



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

1999

Damage control and log taking JAVA applications for shipboard wireless LANs

Sayat, Hanceri.

Monterey, California: Naval Postgraduate School

<http://hdl.handle.net/10945/13470>

Downloaded from NPS Archive: Calhoun



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

**DAMAGE CONTROL AND LOG TAKING JAVA
APPLICATIONS FOR SHIPBOARD WIRELESS LANs**

by

Hanceri Sayat

December 1999

Thesis Advisor:
Second Reader:

Xiaoping Yun
C. Thomas Wu

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 4

20000302 090

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE December 1999	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE DAMAGE CONTROL AND LOG TAKING JAVA APPLICATIONS FOR SHIPBOARD WIRELESS LANs			5. FUNDING NUMBERS	
6. AUTHOR(S) Sayat, Hanceri				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT Damage control communications and watchstander log taking practices need to be improved for onboard submarines and ships. Currently, damage control rely on a slow, error prone process involving sound powered telephone talkers and grease pencil annotated white board. Also log taking practice suffers from similar problems. Logs are taken on paper forms, corrected daily, and filed in cabinets. Wireless network and mobile computing devices can be a solution to improve the efficiency of these practices along with some others. In this thesis, a distributed Java prototype software is developed to utilize the benefit of an onboard Intranet utilizing wireless LANs and pen-based handheld computers. For both practice areas, data could be entered into a handheld computer and then wirelessly transmitted to a database server. Those data can be used by powerful main platforms and different supervisors can reach it any time in parallel. An applet and a servlet program modules are created to provide small, user friendly, platform independent electronic forms. Since handheld computers have some limitations like screen dimensions, computing power, and Java Virtual Machine, features of these software approaches are tested on a few different handheld computers to find the best software approach and computer product.				
14. SUBJECT TERMS Wireless Local Area Network, mobile computing, Java, pen-based computing			15. NUMBER OF PAGES 235	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**DAMAGE CONTROL AND LOG TAKING JAVA APPLICATIONS FOR
SHIPBOARD WIRELESS LANs**

Hanceri Sayat
First Lieutenant, Turkish Army
B.S., Turkish Military Academy, 1992

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

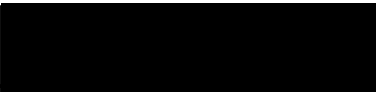
**NAVAL POSTGRADUATE SCHOOL
December 1999**

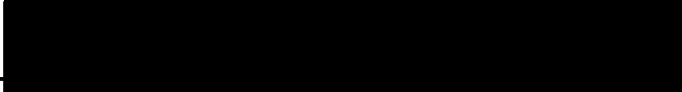
Author:


Hanceri Sayat

Approved by:


Xiaoping Yun, Thesis Advisor


C. Thomas Wu, Second Reader


Jeffrey B. Knorr, Chairman
Department of Electrical and
Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK.

ABSTRACT.

Damage control communications and watchstander log taking practices need to be improved for onboard submarines and ships. Currently, damage control rely on a slow, error prone process involving sound powered telephone talkers and grease pencil annotated white board. Also log taking practice suffers from similar problems. Logs are taken on paper forms, corrected daily, and filed in cabinets. Wireless network and mobile computing devices can be a solution to improve the efficiency of these practices along with some others.

In this thesis, a distributed Java prototype software is developed to utilize the benefit of an onboard Intranet utilizing wireless LANs and pen-based handheld computers. For both practice areas, data could be entered into a handheld computer and then wirelessly transmitted to a database server. Those data can be used by powerful main platforms and different supervisors can reach it any time in parallel. An applet and a servlet program modules are created to provide small, user friendly, platform independent electronic forms. Since small computers have some limitations like screen dimensions, computing power, and Java Virtual Machine, features of these software approaches are tested on a few different handheld computers to find the best software approach and computer product.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION.....	1
A. MOTIVATION.....	1
B. OBJECTIVES.....	2
C. ORGANIZATION OF THE STUDY.....	3
II. BACKGROUND ON WIRELESS LANs.....	5
A. GENERAL.....	5
B. LAN TECHNOLOGY.....	6
C. WIRELESS LAN.....	12
1. Application Areas.....	13
2. Wireless LAN vs. Conventional LAN.....	14
3. Advantages of WLAN	15
a. Mobility.....	15
b. Installation Speed and Simplicity.....	15
c. Installation Flexibility.....	15
d. Reduced Cost-of-Ownership.....	15
e. Scalability.....	15
4. Disvantages of WLAN	16
a. Higher Initial Cost.....	16
b. Slow Throughput Rate.....	16
c. Higher Interface.....	16
d. Security.....	17

e. Industry Standard.....	17
III. JAVA ARCHITECRURE	19
A. GENERAL	19
1. Simple.....	22
2. Object-Oriented.....	22
3. Platform Independent.....	23
4. Secure.....	23
5. High Performance.....	23
6. Multi-Threaded.....	24
7. Distributed.....	24
B. APPLETS.....	25
1. Taking Advantage of the Applet API.....	25
2. Applet Limitations.....	27
3. Applet Security.....	27
4. Security Restrictions.....	28
C. SERVLETS.....	29
1. Taking Advantage of the Servlet API.....	32
2. The advantages of using servlets.....	32
a. Platform Independent.....	32
b. High Performance.....	33
c. Extensibility.....	33
d. Easier Development.....	33
e. Better Error Recovery.....	34

f. Portability.....	34
g. Modularity.....	34
3. Servlet Futures.....	35
D. A DETAILED CONTRAST BETWEEN A JAVA APPLET AND A JAVA SERVLET.....	35
IV. COMMERCIAL TECHNOLOGIES FOR SHIPNET.....	37
A. GENERAL.....	37
B. WLAN PRODUCTS.....	37
1. Lucent WaveLAN.....	37
2. Proxim RangeLAN2.....	42
C. PORTABLE COMPUTER PRODUCTS.....	44
1. Amity VP.....	45
2. Cassiopedia PA-2400.....	47
3. HPW-600ET.....	49
4. Via II Wearable Computer.....	51
D. SHIPBOARD TESTING.....	53
1. Test One.....	53
2. Test Two.....	54
3. Test Three.....	55
4. Test Four.....	56
V. PROTOTYPE SHIPNET SOFTWARE APPLICATIONS.....	57
A. GENERAL.....	57
B. GRAPHICAL USER INTERFACE (GUI) POLICIES.....	59

C. DAMAGE CONTROL CLIENT APPLET.....	60
D. DAMAGE CONTROL CLIENT SERVLET.....	64
E. LOG TAKING CLIENT APPLET.....	68
F. LOG TAKING CLIENT SERVLET.....	71
VI. CONCLUSIOS AND RECOMMENDATIONS.....	75
A. SUMMARY.....	75
B. FURTHER RESEARCH.....	76
C. CONCLUSION.....	78
APPENDIX A. DAMAGE CONTROL CLIENT APPLET CODE.....	79
APPENDIX B. UTILITY CLASSES.....	107
APPENDIX C. DAMAGE CONTROL CLIENT SERVLET CODE.....	113
APPENDIX D. LOG TAKING CLIENT APPLET CODE.....	131
APPENDIX E. LOG TAKING CLIENT SERVLET CODE.....	179
LIST OF REFERENCES.....	209
INITIAL DISTRIBUTION LIST.....	211

LIST OF FIGURES

1. Typical LAN Structure	7
2. OSI Reference Model.....	9
3. Infrastructure WLAN.....	13
4. Adhoc Networking.....	14
5. The Java Architecture.....	19
6. The Java Stack.....	20
7. A server can support more than one servlet.....	29
8. Application Interaction.....	31
9. WaveLAN IEEE PC Card - WEP (Silver).....	39
10. Lucent WavePoint II Access Point.....	40
11. Lucent WavePoint II Range Extender.....	41
12. WaveLAN IEEE PC Card - WEP (Bronze).....	42
13. RangeLAN2 7510/20 Ethernet Access Point.....	43
14. Rangelan2 Dipole and Flexible Snap-On Antenna.....	44
15. Amity VP Portable Computer.....	46
16. Cassiopedia PA-2400.....	48
17. Hitachi's HPW-600ET Tablet.....	50
18. VIA II Wearable Computer (Flex).....	52
19. Test One Result.....	54
20. Test Two Result.....	55
21. Test Three Result.....	56

22. DC Client (Fire) Applet Scene Page.....	62
23. DC Client (Fire) Applet Central Page.....	62
24. Dc Client (Fire) Wacthstander Page.....	63
25. DC Client (Fire) Staging Page.....	63
26. DC Client (Fire) Atmosphere Control Page.....	64
27. DC Client Servlet Menu Page.....	65
28. DC Client Servlet Casualty Scene Page.....	67
29. DC Client Servlet Feedback Page.....	67
30. Log Taking Client Applet.....	69
31. Log Taking Detectors Page.....	70
32. Log Taking Atmosphere Control Page.....	70
33. Log Taking Menu Page.....	72
34. Log Taking AEF Log Page.....	72
35. Log Taking Retrieved Data Page.....	73
36. Log Taking Feedback Page.....	73

LIST OF TABLES

1. Wireless vs. Wired LAN	17
---------------------------------	----

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ABBREVIATIONS, ACRONYMS, AND SYMBOLS

AEF	Auxiliary Electrician Forward
AP	Access Point
API	Application Programmer Interface
BPSK	Biphase Shift Keying
CDMA	Code Division Multiple Access
CGI	Common Gateway Interface
CIC	Communication Intelligence Corp.
COTS	Commercial Off The Shelf
CPU	Central Processing Unit
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DC	Damage Control
DCC	Damage Control Central
DFIR	Diffused Infra-Red
DSSS	Direct Sequence Spread Spectrum
ETSI	European Telecommunications Standards Institute
FCC	Federal Communication Commissions
FDDI	Fiber Distributed Data Interface
FDMA	Frequency Division Multiple Access
FHSS	Frequency Hopped Spread Spectrum
FTP	File Transfer Protocol
GFSK	Gaussian Frequency Shift Keying

GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hyper Text Transport Protocol
HRS	Handwriter Recognition System
ISA	Industry Standard Architecture
ISM	Industrial, Scientific, and Medical
IEEE	Institute of Electrical and Electronics Engineers
JDBC	Java Database Connectivity
JDK	Java Development Kit
JIT	Just-In-Time compiler
JVM	Java Virtual Machine
LAN	Local Area Network
LCD	Liquid Crystal Display
MAC	Media Access Control
NAVSEA	Naval Sea Systems Command
ODBC	Open Database Connectivity
OS	Operating System
OSI	Open Systems Interconnection
PC	Personnel Computers
PCMCIA	Personal Computer Memory Card International Association
PDA	Personal Digital Assistant
QPSK	Quadrature Shift Keying
RMI	Remote Method Invocation

RF	Radio Frequency
SQL	Structured Query Language
SWLAN	Submarine Wireless LAN
TCP/IP	Transmission Control Protocol /Internet Protocol
TDMA	Time Division Multiple Access
URL	Universal Resource Identifiers
USB	Universal Serial Bus
VGA	Video Graphics Array
WAN	Wide Area Network
WLAN	Wireless Local Area Network
WLIF	Wireless LAN Interoperability Forum
WWW	World Wide Web

THIS PAGE INTENTIONALLY LEFT BLANK

THESIS DISCLAIMER

The reader is cautioned that the computer programs developed on this research may not have been exercised for all case of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

Java is a registered trademark of Sun Microsystems.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGEMENT

I would like to thank Professor Xiaoping Yun for his guidance and assistance through the development and completion of this project. He initiated my interest in the topic of pen based portable computers and provided extensive insight and direction in completing this thesis. I am indebted to Professor C. Thomas Wu's assistance in proofing my thesis.

I would also like to thank the contributions of the Wireless Network Research Group at NPS. This includes USN Lieutenants Mark Matthews, Kurt Rothenhaus, and William Wilkins. Lieutenant Rothenhaus provided an initial implementation to the pen based portable processor software applications on a wireless LAN. This helped me a lot to begin my project more smoothly, thanks buddy.

Finally, I need to thank my family for their support from thousands of miles away and my wife, Didem, and my son, Yavuz Bahadir. They each made large personal sacrifices during my work on this thesis. Without their love and support I couldn't have completed this project. I love you each very much.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION

Although wireless networking has rapidly grown in recent years, wireless communications and mobile computing are still in an evolving stage. Every day more and more applications are found that can benefit from wireless networks. These wide-range applications vary from home and small office uses to military uses. Wireless communication is the most important technology behind mobile computing.

Mobile computing will play an important role for the future of military forces. Every level of the combat readiness and effectiveness depends more on electronic systems and uses more computing power than ever before. Parallel to this, more compact, powerful, reliable and faster systems are needed. The main purpose is to communicate the information faster and more efficiently.

To provide the users with flexible and robust communication infrastructure to transmit their messages, highly portable battery-powered computers with wireless communication capability are an obvious solution. The pace of technological advancement in microcomputers is expanding very quickly. Palmtop and wearable mobile computers are becoming cheap, powerful and dependable. So we need to take a serious look at adapting off-the-shelf commercial microcomputers and platform independent software solutions.

Since ships have limited personnel assets, it is important to increase the productivity of every crewmember aboard. The NAVSEA New Attack Submarine (NSSN) program initially identified two areas, damage control (DC) communications and

watchstander logs, for productivity improvement by deploying wireless local area networks (WLANs) onboard submarines. Accurate, timely communications between the casualty scene, different stations around the ship, and Damage Control Central (DCC) have always been of the utmost importance when combating shipboard casualties. Current damage control communications practices inboard submarines rely on a slow, error prone process involving sound powered telephone talkers and a grease pencil annotated white board. The transcription of the status information to the white board is limited to the rate and accuracy of a single person receiving and writing the voice communications and is only available to those personnel in view of the white board. There is a great need to improve DC communications. The current practice of watchstander log taking has similar needs for improvement. Logs on today's submarines are taken on paper forms, collected daily, and stapled with other watchstander logs in a large bundle. This bundle is reviewed sequentially by responsible supervisors and filed in cabinets. It is usually never looked at again. The biggest problem of this procedure is that it discourages trend analysis, if a trend analysis is to be performed, data must be either hand plotted or entered into a computer [Ref. 1].

B. OBJECTIVES

This thesis investigates mobile computing and wireless networking in three ways. First, it provides a study of wireless LANs that have potential of providing transparent, reliable communication for Navy ships' or submarines' damage control and log taking systems. It defines the requirements that should be met and it determines the technology

alternatives, products and configurations providing a solution to the required wireless LAN. The study focuses on mobile computers and communication devices. Mobile computers and wireless LANs can assist in the integration of information. This may not only be considered as tools for a command and control hierarchy, but rather as a means of helping each personnel accomplish their duties.

Second, applet and servlet software modules are created for damage control and log taking systems. These software approaches provide a good interaction mechanism between a user and a central database by using distributed Java software. It also demonstrates the easy and effective use of pen-based miniature computers with Java applications over wireless network for Shipnet.

This study then creates a test environment with this prototype software on Shipnet. The goals are to see the performance of wearable and hand held computers, speed and robustness of those software approaches and effectiveness of user interface.

C. ORGANIZATION OF THE STUDY

Chapter II offers the reader a brief introduction to wireless local area networks.

Chapter III provides an overview of Java applets and servlets. It also discusses the pros and cons of both applications in terms of distributed computing, and server and client side approaches.

Chapter IV focuses on the Commercial-off-the-shelf products used in the Submarine Wireless LAN (SWLAN).

In Chapter V, the Java applet and servlet programs written for damage control (DC) and watchstander logs will be presented in detail. A comparison of applet and servlet approaches is carried out.

Finally, Chapter VI summarizes this thesis research and offers a road map for future researches.

II. BACKGROUND ON WIRELESS LANs

A. GENERAL

Over the past decade, the field of wireless communications has experienced an unprecedented growth. This growth will be more visible in the next decade. The deployment of wireless networks has already begun to change the concept of communication and information processing for business, military, and private applications. Currently, there is an enormous shift in the computer industry toward integration of high performance distributed computing and portable devices in a mobile computing environment. Furthermore, military agencies are developing personal wireless devices for use in tactical environments, as well as portable devices providing extensive computational power for sailors and soldiers. These factors lead to a conclusion that communication shifts dramatically from wired based systems to an era of networks largely based on wireless techniques.

The new concept of wireless LANs is a direct result of an increasing demand for portable information sharing in a wireless networking environment from computer users. WLAN offers the user a degree of mobility never thought possible in a conventional LAN environment. Although reliable links of high data-rates can be achieved using wired technologies, such as coaxial cable and optical fiber, WLAN has the unique advantage in providing a simple and cost effective solution when it comes to reconfiguration, maintain and upgrade, which is very difficult and costly for conventional LANs [Ref. 2].

Currently, spread spectrum radio technique is the most widely used transmission technique for WLANs. It was initially developed by the military to avoid jamming and eavesdropping of the signal. When used in wireless LAN applications this is done by spreading the signal over a range of frequencies, which consists of the Industrial, Scientific, and Medical (ISM) bands of the electromagnetic spectrum. This band's frequencies range between 902-928 MHz and 2.4 - 2.484 GHz, which do not require Federal Communication Commissions (FCC) license. In the Shipnet project, the bandwidth and data transfer rates are sufficient for the application's characteristic and work load. Although current rates are satisfactory for our goal, better performances may be needed for future application like multimedia applications, online training, etc. For further improvements, manufacturers are still trying to come up with innovative solutions for interfacing between wired and wireless systems. So the band is moving towards the millimeter wave frequencies (mm-wave band) to get better achievable data rates and to avoid the congested RF band [Ref. 3].

B. LAN TECHNOLOGY

A LAN is a network of interconnected workstations sharing the resources of a single processor or server within a relatively small geographic area, Figure 1. Typically, this might be within the area of a small office building. Usually, the server has applications and data storage that are shared in common by multiple workstation users. A local area network may serve as few as four or five users or, in the case of Fiber Distributed Data Interface (FDDI), may serve several thousand.

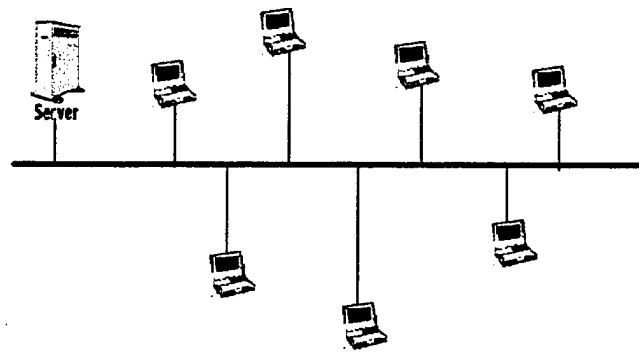


Figure 1. Typical LAN Structure

There are many different types of LANs. For instance, Ethernet is the most common one. Most Apple Macintosh networks are based on Apple Talk network system, which is built into Macintosh computers.

The following characteristics differentiate one LAN from another:

Topology : The geometric arrangement of devices on the network. For example, devices can be arranged in a ring or in a straight line.

Protocols: The rules and encoding specifications for sending data. The protocols also determine whether the network uses a peer-to-peer or client/server architecture.

Media : Devices can be connected by twisted-pair wire, coaxial cables, or fiber optic cables. Although today, twisted pair continues to dominate much of the LANs, the shift to fiber optic cables for specialized high-speed applications is noticeable.

The typical LANs provide a logical and efficient choice for most networks where the users operate from a fixed desk in a standard office environment. LANs are typically identified by the following properties:

1. Multiple systems attached to shared medium,
2. High total bandwidth ,
3. Low delay ,
4. Low error rate,
5. Broadcast/Multicast capability,
6. Limited geography (1-2 km) ,
7. Limited number of stations ,
8. Peer relationship between stations,
9. Confined to private property.

The most common forms of LAN are those described by the Institute of Electrical and Electronics Engineers (IEEE) standards. This standard describes operation up to and including Open Systems Interconnection (OSI) level 2. Individuals may build what they like on top of these basic protocols. A common set of higher level protocols is called Transmission Control Protocol/Internet Protocol (TCP/IP) which provides OSI levels 3 and 4 functionality [Ref. 4]. On top of them there are some other protocol layers which are shown in Figure 2. At the lowest level the IEEE 802 specifications are split four ways corresponding to three different but common LAN structures.

They are :

1. **802.3** : This is the well-known Ethernet. All stations are connected to a common transmission medium. The medium is organized as a bus, possibly with branches. Each station transmits whenever it wishes. This may result in two stations

transmitting simultaneously. The protocol describes how such collisions are detected and resolved.

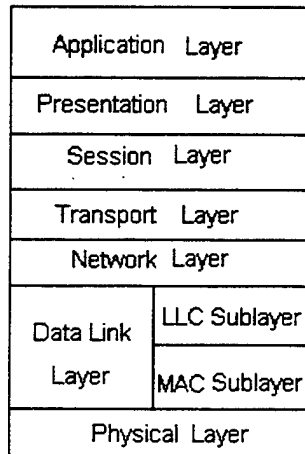


Figure 2. OSI Reference Model

2. **802.4** : This is a token bus system. The medium is organized in the same way as for IEEE **802.3** but the stations are organized as a logical ring. Only a station possessing "the token" may actually transmit. There are procedures defined in the protocol for the addition of a station, deletion of a station and error detection and recovery.

3. **802.5** : This is a true ring. The medium is organized as a physical ring with access control protocols similar to IEEE 802.4. Rings may also be organized as slotted rings (Cambridge Ring) and register insertion rings (IBM Series/1), neither of these are associated with IEEE 802.5.

4. **802.11** : The IEEE 802.11 provides a standard for wireless connectivity for fixed, portable, and moving stations within a local area. It is similar to the 802.3 standard for Ethernet wired LANs. This standard allows interoperability between different vendors' equipment. With 802.11, customers can mix and match their equipment for an optimal product mix to support all of their requirements. The mission and the scope for this standard are to develop a MAC and physical specification for radio and for infrared.

The MAC under 802.11 is composed of several functional blocks. These include mechanisms to provide control on a variety of physical layers. The functions within the MAC are independent of data rates or physical characteristics.

The fundamental access method of the 802.11 MAC is known as Carrier Sense Multiple Access with collision avoidance (CSMA/CA). CSMA/CA works by a "listen before talk scheme." This means that a station wishing to transmit must first check the radio channel to determine if another station is transmitting. If the medium is not busy, the transmission may proceed. The CSMA/CA scheme implements a minimum time gap between frames for a given user. Once a frame has been sent from a given transmitting station, that station must wait until the time gap is up to try to transmit again. Once the time has passed, the station selects a random amount of time (called a backoff interval) to wait before "listening" again to verify a clear channel when to transmit. If the channel is still busy, another backoff interval is selected which is less than the first. This process is repeated until the waiting time approaches zero and the station is allowed to transmit. This type of multiple access ensures effective channel sharing while avoiding collisions.

This scheme also allows automatic medium sharing between several devices with compatible PHYs. This access method is attractive because it provides both spectral efficiency and asynchronous data transfer. Frequency Division Multiple Access (FDMA) and Code Division Multiple Access (CDMA) schemes would not be enough because they require bandwidth used by the modulation scheme. Strict Time Division Multiple Access (TDMA) would not work well because it requires synchronization. Thus CSMA/CA, which may be thought of as a version of TDMA, is better suited to this application.

The PHY under 802.11 includes diffused infra-red (DFIR), direct sequence spread spectrum (DSSS), and frequency hopped spread spectrum (FHSS). Both spread spectrum techniques are used in the 2.4 GHz band because of wide availability in many countries and lower hardware costs in comparison to the higher microwave frequencies.

The IEEE 802.11 supports DSSS for use with Biphase Shift Keying (BPSK) modulation at a 1 Mbps data rate, or Quadrature Shift Keying (QPSK) modulation at a 2 Mbps data rate. The general band plan consists of five overlapping 26 MHz sub-bands centered at 2.412, 2.427, 2.442, 2.457, and 2.470 GHz. This scheme is used in an attempt to combat interference and selective fading [Ref. 4].

FHSS is supported under 802.11 with Gaussian Frequency Shift Keying (GFSK) modulation and two hopping patterns with data rates of 1 Mbps and 2 Mbps. Under this scheme, the band is divided into 79 sub-bands with 1 MHz bandwidth each. Each sub-band is subject to a minimum rate of 2.5 hops/sec using any of three possible hop patterns (22 hops in a given pattern). The minimum hop rate ensures that each packet sent could be transmitted in a single hop so that missed information could be recovered in another

hop. This allows an effective frequency diversity that provides excellent transmission characteristics [Ref. 2].

C. **WIRELESS LAN**

When wireless LANs were introduced on the market at the end of last decade, most products were designed to be used together with stationary desktop computers. Their main objective was to offer flexible and cost-effective alternatives to the LAN cable. The basic infrastructure of a wireless LAN is given in Figure 3. Aside from saving the often high costs of installing and maintaining the cable, another benefit of wireless LANs was the speed and ease with which new LANs could be created or new computers added.

These benefits are of course still valid today, but mobile computing is changing at an amazing pace. As mentioned before, advances in low power microprocessors, display quality and battery technology are resulting in extremely powerful notebook PCs. Consumer electronics and personal computers are merging to create a class of even smaller computing products, the PDAs. With the integration of telephone and multimedia technology in these devices, intelligent terminals for telecommunication services were created.

Users need the freedom to move around the workplace unhindered by network cable and power cords. At the same time, they require timely access to all of the information. This type of mobile computing has placed an entirely new set of demands on LANs, such as higher data-rates, more reliable services, and shorter antennas.

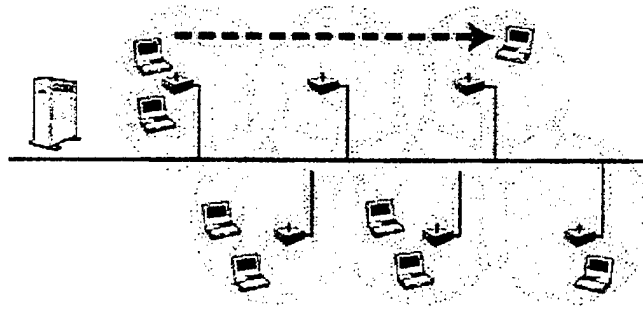


Figure 3. Infrastructure WLAN

1. Application Areas

Wireless LANs have been used in a number of application areas. In addition to office environments, wireless LANs are today used in hospitals to provide personnel with the possibility to gain access to information such as patient records while on the move. Similar needs are expressed by personnel working at airports, construction areas, universities, and the military etc. One application area for wireless LANs is to bridge between cabled LANs in different buildings. Ad-hoc networking is another application, which is expected to take advantage of the wireless connection.

Ad-hoc networking is what happens when two or more workstations are connected to form a network at the time it is needed. That network exists only as long as there are stations in it. Figure 4 shows an ad-hoc WLAN structure.

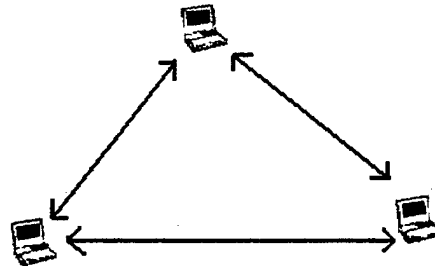


Figure 4. Ad-hoc Networking

2. Wireless LAN vs. Conventional LAN

Conventional LANs provide fast and reliable connection for desktop computer to share information via wire, coaxial cable or optical fibers. The WLAN serves the same purpose as that of a wired or optical LAN, carrying information among the devices attached to the LAN. The WLAN is much more flexible with the lack of physical cabling to tie down the location of a node on a network. As opposed to the large amount of labor required adding or moving the cabling in wired network, WLAN offers a solution to overcome the difficulties associated with installing, maintaining and upgrading networks in typical building environments. Wireless connections enable portable computers to retain their portability without sacrificing the advantages of being connected to a network [Ref. 5] .

3. Advantages of WLANs

Wireless LANs offer the following productivity, convenience, and cost advantages over traditional wired networks. A concise comparison between a wired and wireless LAN is shown in Table 1.

a. Mobility

Wireless LAN systems can provide LAN users with access to real-time information anywhere within their organization. This mobility supports productivity and service opportunities not possible with wired networks.

b. Installation Time and Simplicity

Installing a wireless LAN system can be fast and easy and can eliminate the need to pull cable through bulkheads, deck plates of a ship.

c. Installation Flexibility

Wireless technology allows the network to go where wire cannot go.

d. Reduced Cost-of-Ownership

While the initial investment required for wireless LAN hardware can be higher than the cost of wired LAN hardware, overall installation expenses and life-cycle costs can be significantly lower. Long-term cost benefits are greatest in dynamic environments requiring frequent moves, changes, and routing of terminals and cabling.

e. Scalability

Wireless LAN systems can be configured in a variety of topologies to meet the needs of specific applications and installations. Configurations are easily changed and range from peer-to-peer networks suitable for a small number of users to full infrastructure networks of thousands of users that enable roaming over a broad area.

4. Disadvantages of WLANs

Although the advantages of the WLANs far more outnumber the disadvantages, the major disadvantages related to wireless LAN can be summarized with the following categories.

a. Higher Initial Cost

Although the initial cost for a wireless LAN is two to three times more than that of a conventional LAN, investors can regain the investment for WLAN after a couple of moves. The cost depends primarily on the number of access points deployed. The number of these access points depend on the coverage region and the number of and type of users to be serviced. The quick capital recovery of wireless LAN relies on two features of wireless LAN. Firstly, wireless LAN discards the installation and maintenance costs and labor of the network. Next, the indirect costs of user downtime and administration overhead are reduced during network relocation period [Ref. 3].

b. Slow Throughput Rate

Airwave congestion (number of users), propagation factors such as range and multipath, the latency and bottlenecks on the wired portion of the WLAN are some of the factors affecting wireless LAN. The most common throughput rate for WLAN is around 1-10 Mbps, while that of a wired LAN can reach up to 100 Mbps.

c. Higher Interface

The products transmitting energy in the same spectrum with WLAN can potentially cause some level of interference. This factor degrades the throughput.

d. Security

In mm-wave band, solid objects such as walls, equipment are transparent to this signals. This fact may cause a security risk for some environment. Coverage areas may need to be measured carefully. Some security measures should be considered.

e. Industry Standard

The IEEE 802.11 specifications were published in 1997. Unfortunately, the standard doesn't specify technology or implementation but only specifications for the physical layer and MAC layer. It needs a lot of work to reach the wired networks standard.

Attribute	Wireless LAN	Wired LAN
Cost	Initial investment in hardware costs is higher Installation costs and maintenance costs can be much lower	Investment cost in hardware is lower Installation and maintenance costs can be significantly higher
Scalability	Network can be simple to complex	Network can be simple to complex
Safety	Very little exposure to radio frequency energy	No exposure to radio frequency energy
Mobility	Provides access to real-time information anywhere	Does not support mobility
Throughput	1- 10 Mbps	10 - 100 Mbps
Integrity & Reliability	Subject to interference	Highly reliable
Simplicity / Ease of Use	No need to pull cable / Set up time is lower / Moves, additions & changes much simpler	Cable required / Set up time is significantly higher
Security	Susceptible to interception Overcome with encryption	Not as susceptible to interception

Table 1. Wireless vs. Wired LAN

THIS PAGE INTENTIONALLY LEFT BLANK

III. JAVA ARCHITECTURE

A. GENERAL

Java is a new high level programming language developed by Sun Microsystems. It is a language made for writing client/server and networked applications, and especially well suited for the Internet. It is object-oriented and has many similarities with the C and C++ languages. But unlike C, Java was designed to be a platform independent program language. Java is compiled to an intermediate byte-code which is executed by a Java interpreter. Therefore, the Java interpreter and a few native code libraries are enough to port Java programs to a new platform. Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (JVMs), exist for most operating systems, including UNIX, Macintosh operating system (OS), and Windows. Java architecture is shown in Figure 5.

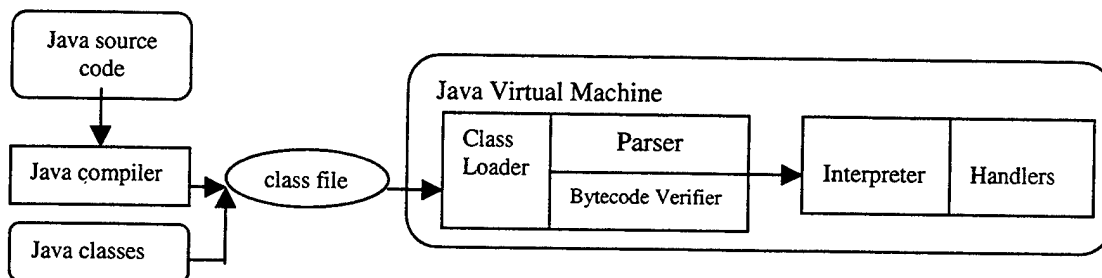


Figure 5. The Java Architecture

Java goes back to 1991, when a group of engineers from Sun Microsystems wanted to design a small computer language that could be used for consumer devices like cable TV switchboxes. Since these devices don't have a lot of power nor a lot of memory,

the language had to be small and generate very tight code. Also, because different manufactures may use different central processing units (CPUs), it was important to make the language architecture neutral. Java was originally called Oak. In 1995, Sun changed the name to Java and modified the language to take advantage of the growing World Wide Web (WWW). Java is two different things. First the language, second the Java Virtual Machine. It is important to keep these two things apart.

The Java language is the means for a developer to write source code. Applets and applications written in Java compile to a form that runs on the Java platform. Java platform can be defined as a software-only platform that runs on top of other hardware based platforms. It is a programming environment developed by Sun Micro Systems which is used to build and deploy Java based software applications by using well-established and proven two components, Java virtual machine and Java API. The platform comes in different versions. Depending on how advanced a JVM is one obtain a certain level of compatibility. The compatibility refers to the amount of Application Programmer Interfaces (APIs) a JVM supports. An API is a package containing classes and methods that may be used as a tool to aid the programmer. The model of the Java stack can be seen in Figure 6.

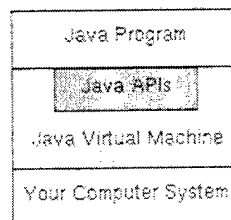


Figure 6. The Java Stack

There are two different kinds of programs that run on the Java Platform:

Applets : Small applications that require a web browser to run. The applet is embedded in a web page and is downloaded automatically from a remote server when the page is accessed. It then runs on the local host. The JVM is built into web browsers.

Applications : Java code does not need a browser and is not automatically downloaded. Applications run when called upon just like in any other programming language. They do need a local JVM to execute the bytecode.

The difference between an ordinary machine (i.e. a physical computer) and the Java Virtual Machine is that the JVM does not exist physically. One can view the JVM as a computer program describing a software processor.

Java applets can be downloaded from a Web server and run on any computer by a Java-compatible Web server, such as Netscape Navigator or Microsoft Internet Explorer. Java is a programming language for distributed applications. It doesn't just allow programmer to add new types of content to Web pages like Netscape and Internet Explorer do. Rather it lets a user add both the content and the code necessary to interact with that content. One no longer needs to wait for the next release of a browser that supports a preferred application format. With Java, a user sends browsers both the content and the program necessary to view the content. For instance let's say crewmembers on a ship are able to search the ship's Log Database. However the database exists on a mainframe system that doesn't speak HTTP (Hyper Text Transport Protocol). Before Java one could hope that some browser implemented a specific database protocol or could try to program some intermediate cgi-bin on OS that can speak HTTP and talk

to the database, not an easy task. With Java when clients want to talk to a database, someone can send them the code they need to do so.

The design requirements of Java are driven by the nature of the computing environments for which software must be developed. As a language Java is

1. Simple

Java has the bare bones functionality needed to implement its rich feature set. It does not add lots of syntactic and unnecessary features. Java was designed to make it a lot of easier to write bug free code without extensive programmer training. The language is strictly typed to catch bugs early. It has automatic garbage collection so that the programmer won't have to worry about memory administration. Like in C language, Java has no pointers and no global variables, which makes the code reading easier. Without pointer arithmetic, a lot of errors and headaches can be avoided.

2. Object-Oriented

Almost everything in Java is either a class, a method or an object. Only the most basic primitive operations and data types (integer, for, while, etc.) are at a sub-object level. As a programmer, this means that one focus on the data in applications and methods that manipulate that data, rather than thinking strictly in terms of procedures.

3. Platform Independent

Java is designed to support applications that will be deployed into heterogeneous networked environments. In such environments, applications must be capable of executing on a variety of hardware architectures. As mentioned before Java programs are compiled to a byte code format that can be read and run by interpreters on many platforms such as Windows 95, Windows NT, UNIX, and Solaris 2.3. All that is needed is a JVM written for that platform which has been done already.

4. Secure

Java is designed to operate in distributed environments, which means that security is very important. Java code can be executed in an environment that prohibits it from introducing viruses, deleting or modifying files, or otherwise performing data destroying and computer crashing operations. Many of the defenses in Java protect the user from untrusted applets. The JVM contains a bytecode verifier to ensure that code does not violate any Java language restrictions. The verifier is activated before the code is executed. It provides extensive compile-time checking, followed by a second level of run-time checking.

5. High Performance

Performance is always a consideration. Java achieves superior performance by adopting a scheme by which the interpreter can run at full speed without needing to check

the run-time environment. Java can be compiled very quickly with a Just-In-Time compiler (JIT) to code that comparable with C++ in speed.

6. Multi-Threaded

Java is inherently a multithreaded language which means that it provides support for multiple threads of execution that can handle multitasking. A single Java program can have many different things processing independently and continuously. This is very valuable in interactive applications.

7. Distributed

With the phenomenal growth of networks, today's developers must think distributed. Applications must be able to migrate easily to a wide variety of computer systems, a wide variety of hardware architectures, and a wide variety of operating system architectures. A Java application/applet may also have its code spread out on more than one computer since the JVM will load the classes it needs when they are actually needed.

Java supports various levels of network connectivity through classes in the package `java.net`. For instance, the `URL` class allows a Java application to open and access remote objects on the Internet.

B. APPLET

Although a short definition of an applet was previously defined, the features of applets need closer scrutiny to understand the importance of it for this project and to compare it with servlet. An applet is a small program designed to run within another application. Applets are useful for network use especially on the Web because, once they are downloaded, they can be executed quickly within the user's browser. More than one applet can exist in a single document, and they can communicate with one another while they work. Java is one of the major languages used for creating Web-based applets.

1. Taking Advantage of the Applet API

The applet API lets a user take advantage of the close relationship that applets have with Web browsers. The API is provided by the `java.applet` package, mainly by the `Applet` class and the `AppletContext` interface. The `java.applet` package provides an API that gives applets some capabilities that applications don't have. For example, applets can play sounds, which other programs can't do yet. Here are some other things that current browsers and other applet viewers let applets do:

- a.* Applets can usually make network connections to the host they came from.
- b.* Applets running within a Web browser can easily cause HTML (Hyper Text Markup Language) documents to be displayed.
- c.* Applets can invoke public methods of other applets on the same HTML page.

d. Applets that are loaded from the local file system (from a directory in the user's CLASSPATH) have none of the restrictions that applets loaded over the network do.

e. Although most applets stop running once the user leaves their page, they don't have to.

Java Applets are limited to certain operations on the browser. On the other hand it has some other capabilities, too. Java applets can use a JDBC (Java Database Connectivity) connection or distributed objects. The Java applet allows more complex user interface options than HTML combined with either Common Gateway Interface (CGI) or Java servlets. However, the Java applet requires a Java-enabled Web browser. Applets have several advantages over HTML.

a. Applets allow local validation of data entered by the user. Local validation of data is possible using HTML combined with JavaScript, but variances in JavaScript implementations make JavaScript difficult to use.

b. An applet can use the database to perform value lookups and data validation. HTML (even if combined with JavaScript) can not do that without invoking a CGI or servlet program and drawing a new HTML page.

c. Once an applet is downloaded, the amount of data transferred between the Web browser and the server is reduced. HTML requires that the server transfer the presentation of the data (the HTML tags) along with the data itself. The HTML tags can easily be 1/4 to 1/2 of the data transferred from the server to the client.

d. Applets allow the designer to use complex Graphical User Interface (GUI) widgets such as grids, spin controls, tab panels and scrollbars. These tools are not available in HTML.

2. Applets Limitations

For security reasons, applets that are loaded over the network have several restrictions. One is that an applet can't ordinarily read or write files on the computer that it's executing on. Another is that an applet can not make network connections except to the host that it came from. Despite these restrictions, applets can do some things that one might not expect. For example, applets can invoke the public methods of other applets on the same page.

3. Applet Security

The goal for the JDK (Java Development Kit) is to enable browsers to run untrusted applets in a trusted environment. The approach is to be conservative at first, and to add functionality when it can be added securely. The intent is to prevent applets from inspecting or changing files on the client file system. Also, the intent is to prevent applets from using network connections to bypass file protections or people's privacy.

JDK provides the basic technology for loading and authenticating signed classes. This enables browsers to run trusted applets in a trusted environment. This does not make obsolete the need to run untrusted applets in a secure way.

4. Security Restrictions

Every browser implements security policies to keep applets from compromising system security. The implementation of the security policies differs from browser to browser. Also, security policies are subject to change. For example, if a browser is developed for use only in trusted environments, then its security policies will likely be much more loose than those described here.

Current browsers impose the following restrictions on any applet that is loaded over the network:

- a.* An applet cannot load libraries or define native methods.
- b.* It cannot ordinarily read or write files on the host that's executing it.
- c.* It cannot make network connections except to the host that it came from.
- d.* It cannot start any program on the host that's executing it.
- e.* It cannot read certain system properties.
- f.* Windows that an applet brings up look different than windows that an application brings up.

Each browser has a `SecurityManager` object that implements its security policies. When a `SecurityManager` detects a violation, it throws a `SecurityException`.

The applets can catch this `SecurityException` and react appropriately. This is known as exception handling, which the Java language is very good at.

C. SERVLETS

According to Sun Microsystems, a servlet is "a standard approach to extending server functionality without the limitations of CGI-based or server-specific approaches". Simply, a servlet is the opposite end of an applet, or a faceless applet. A servlet can almost be thought as a platform-independent 100% Pure Java server-side applet. Servlets run inside the Web server just as applets run inside the Web browser. Servlets gives request /response-oriented services and extend them in some manner. It executes on the Web or HTTP server in response to requests from a Web browser, performing tasks traditionally handled by CGI programs. A sever can run more than one services at the same time. An example is seen in Figure 7. Web servers that can support servlets include: Apache, Netscape's Fast Track and Enterprise Servers, Microsoft's IIS, O'Reilly's WebSite, and JavaSoft's Java Web Server.

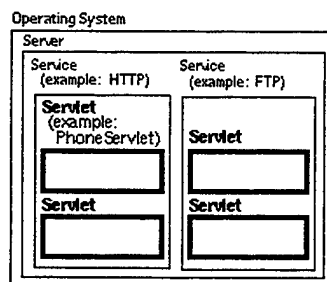


Figure 7. A server can support more than one servlet

They can be used to extend the capabilities of the HTTP or Web servers in a variety of ways with minimal overhead, maintenance, and support. Because servlets are Java byte codes that can be downloaded or shipped across the network as applets, they are truly "Write Once, Run Anywhere". Therefore, they provide a standard alternative to CGI programming that enhances the performance, security and implementation.

A plain HTML document that a Web server retrieves is static. A servlet, on the other hand, is executed for every request so that it can output dynamic information. For example, a browser generates a request to a server for a document that contains dynamic information. The server examines the request and maps it to a particular servlet. It then invokes the servlet which creates the document with dynamic content and returns it to the client. Figure 8 shows the application interaction of a running servlet.

The Web server software can use a Java Virtual Machine to run the servlet and generate a HTML page. The servlet takes input from a HTML page containing HTML input tags, processes it, and returns a HTML page with the results. Java servlets can be thought of as CGI programs in some respects. Both servlets and CGI execute on the server in response to HTML input. Thus both are limited to HTML and JavaScript's capabilities. However, Java servlets are much more powerful than CGI programs.

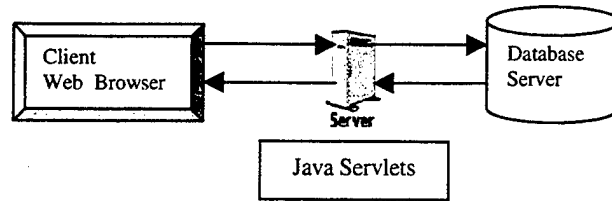


Figure 8. Application Interaction

Servlets can do more than just return documents. Once the HTTP connection is opened the client and servlet can speak a custom protocol on the connection. Servlets can be thought of as server side applets. The difference is that servlets are faceless objects without a user interface. Servlets can be long-lived. A new servlet does not have to be created for every request. The server just needs to call the servlet method that can respond to the request.

Servlets can be dynamically loaded in a running server. They can be loaded from the local disk or from the network. Agents are programs that can roam a network, interact with the host, gather information, and come to the host where they originated. Servlet technology is an important step forward in achieving this goal.

Servlets make use of the Java standard extension classes in the packages `javax.servlet` (the basic Servlet framework) and `javax.servlet.http` (extensions of the Servlet framework for Servlets that answer HTTP requests). Since Servlets are written in the highly portable Java language and follow a standard framework, they provide a means to create sophisticated server extensions in a server and operating system independent way.

1. Taking Advantage of the Servlet API

The Java servlet API is the first standard extension to Java for interfacing servlets with information servers, such as HTTP or Web servers. The servlet API contains methods for initializing a servlet, processing the request, getting servlet information, and destroying the servlet. The servlet API is simple, flexible, and stable. Application developers can write platform independent servlets and expect them to run on any server that supports the servlet API.

Servlets have full access to the various Java APIs and to third-party component classes, making them ideal for use in communicating with applets, databases, and Remote Method Invocation (RMI) servers. The servlet is a very new application compared with the applet. But the servlet API is very simple. As a consequence, it is pretty stable. The Servlet API, like Java itself, is being constantly improved by JavaSoft. There will be a few things to add in the next version of the Servlet API.

2. The advantages of Using Servlets

Sun is aiming to make servlets the new Web server programming paradigm. Servlets are persistent, platform independent, and incorporate all sorts of advanced features including security, easy database access, and much easier integration with Java applets.

a. Platform Independent

Servlets can run on any platform without recompilation or rewriting, but CGI and server extensions written in other high-level languages such as C and C ++ are not as portable.

b. High Performance

Servlets only need to be loaded once, while CGI programs need to be loaded for each request. The servlet `init()` method allows programmers to perform resource intensive actions (such as database connections) at startup and reuse them across servlet invocations. Instead of connecting to a database thousands of times a day, a user connects once.

The built-in thread support of Java technology not only makes servlet development easier, it also improves performance. Because CGI programs create a separate process for each request, a heavy traffic can easily occur on the server with too many simultaneous processes. Servlets, on the other hand, use a single process, multithreaded model with comparatively little cost in handling requests

c. Extensibility

Java is a robust, well-designed and fully object-oriented language. Specialized Java libraries, development tools and database drivers are becoming available all the time, and servlets can utilize Java code from any source.

d. Easier Development

Using Java to develop CGI type programs is much easier and more bug-free than C, Perl or any other programming languages. Developers have all the benefits of Java as a CGI-type language. Java simply has better support for building web applications, without the headaches and risks of pointers and memory allocations. Overall, it's an easier and friendlier development environment. By using the Servlet API, the developer doesn't have to worry about the inner working of the server. Data, server headers, and cookies, etc, can all be handled by the servlet's underlying classes.

Additionally, because the servlets are written in Java, one can move them from one servlet to another, and does not worry about the operating system or the type of server.

e. Better Error Recovery

From the developer view, perhaps the biggest advantage of servlet technology is the way it handles itself during runtime errors. With CGI, if a program crash occurs, the users only get the server error screen, they can't tell where the error occurs and what kinds of error are. With the very handy exception handling of servlets, developers are able to provide the user with detailed instructions on who to contact, along with a stack trace showing which line of code caused the problem.

f. Portability

The Internet is a dynamic environment where things can change quickly. The Java platform offers the flexibility to react to this changing environment, the ability to run on multiple operating systems, and even the ability to move from a client-side to server-side implementation. One can develop a complex server-side application without restricting it to a particular hardware platform. Today, one application servlet can reside on the Unix platform and then one can later move it to a Windows NT platform. This migration can take place without the headaches associated with porting code and without the need to recompile Java servlets.

g. Modularity

Supporting the idea of modularity, each servlet can perform a specific task and then they can be linked together. Servlet can talk to each other. This mechanism is known as servlet chaining.

3. Servlet Futures

Servlets are powerful tools of the Java technology, which have recently gained attention. Many servlet development companies believe that the current definition is too narrow. An ideal servlet should be considered as "mini-servers" and does not need to be linked to a Web server handling web transactions, database access and vertical applications. In a short period, servlets will become increasingly popular. This area will gain the most "real" usage of Java in the market.

D. A DETAILED CONTRAST BETWEEN AN APPLLET AND A SERVLET

Basically, a servlet is the opposite end of an applet. A servlet can almost be thought of as a server-side applet. Servlets run inside the Web server in the way that applets run inside the Web browser. The browser can submit a request to execute a servlet directly. It can be stand-alone in terms of its actions, as a browser can request an applet directly.

Java Servlets often work hand in hand with applets, providing a high degree of interactivity and dynamic updating. A scheduling applet, for example, could interact with a Java Servlet to provide a common calendar for a group of users. The applet could provide interactive features, such as a scrolling calendar view, that are updated by the Java Servlet dynamically.

Applets:

1. Run on the users machine
2. Strong GUI, graphics, etc.

Servlets:

1. Run on the server
2. Good at generating dynamic HTML pages

For example, if we are talking about data retrieval, an applet is better if a powerful user interface is necessary. An HTML form is inadequate for getting the user's input. While a servlet is better if the display of the output is more important. Of course both can be used.

Currently, Servlets may be the best possible use of Java for handheld units which doesn't support a JVM. There is more flexibility with Dynamic HTML to create more powerful client-side applications. But Java as a server-side tool is very powerful. The good thing is that the power of Java can be realized right away. The browser support is not a major issue any more. Servlet solves the JVM versioning problems that may be faced with client-side Java. When a new version of JDK comes out, there is nothing to worry about Java support on the browser.

Some detailed implementation issues related to applet and servlet applications will be made clear in the following chapters.

IV. COMMERCIAL TECHNOLOGIES FOR SHIPNET

A. GENERAL

Developing products for the commercial telecommunication's marketplace is a fast growing area. Wireless products are currently one of the most important parts of this hot market. A lot of different companies offer a variety of hardware and software connectivity products and services that help consumers take advantage of the opportunities presented by wireless computing. The wireless data market provides compelling applications and products to wireless users that increase the productivity and efficiency. Mobile computer market consisting PDAs, hand held and wearable computers, CE devices shows a similar increasing trend of productivity like wireless technology does. Everyday a new product is introduced to the market with smaller dimensions, and more powerful hardware and software specs.

For Shipnet project, COTS (Commercial off the shelf) wireless LAN and computer products were investigated and tested for onboard use. In the scope of this research, features of Lucent's WaveLan and Proxim's RangeLAN2 wireless products, and Mitsubishi's Amity, Casio's Casiopedia, Hitachi's SuperH, and VIA's Flex mobile computers are investigated.

B. WLAN PRODUCTS

WLANs allow users to stay connected to their usual network while retaining their freedom of movement. WLANs permit systems to be added, moved and removed easily

and flexibly. One of the basic parts of a WLAN is mobile user units with light radio and interface cards or a Personal Computer Memory Card International Association (PCMCIA) card packaging. The other one is an access point which is a larger external radio unit with some additional equipment such as range extender, and different kinds of antennas. The access point provides connectivity to an Ethernet segment with transparent bridging functionality. Within a cell, the system replaces the Ethernet physical and MAC layer functionality in a way that is fully transparent to the user.

In integrating WLANs to the wired networks successfully, MAC bridges have a critical role. They receive the data from one LAN and retransmit to the destination on another LAN. The bridge operates at the data link layer of the OSI model. It is protocol independent but media dependent. The bridge can manage the traffic travelling to various segments of the network. A MAC layer bridge keeps the port number that the packets form, a particular source address, so it will know where to send the reply packets. If the bridge currently knows the destination port number, it will directly forward the packets to the destination. Otherwise, It will broadcast packets to all available ports. By propagation through the network, packets reach their destination [Ref. 7].

Among commercial wireless products, Lucent and Proxim are those providing high data rates and good connectivity properties. They are utilized in the prototype Shipnet model.

1. Lucent WaveLAN

WaveLAN is a family of products using Radio Frequency (RF) technology and based on the IEEE 802.11 standard for wireless LANs. Products include the WavePOINT

fixed wireless access points and bridges, network interface cards for computer devices, and antenna systems.

The card shown in Figure 9 can be installed into a mobile computer's Type II PC card slot or into a desktop PC via an ISA-bus (Industry Standard Architecture) PCMCIA adapter. These cards communicate with a Lucent WavePoint II Access Point shown in Figure 10, which acts as a hub. The WavePoint also can connect one or two wireless LANs to a standard wired Ethernet LAN using its two PC card sockets and standard 10Base-T RJ-45 jack. According to Lucent, mobile users can roam seamlessly from one Access Point's coverage area (a cell) to another. During the tests, there were only momentarily slight drops which did not affect the smooth transition from one access point to the other, which confirms Lucent assertion.

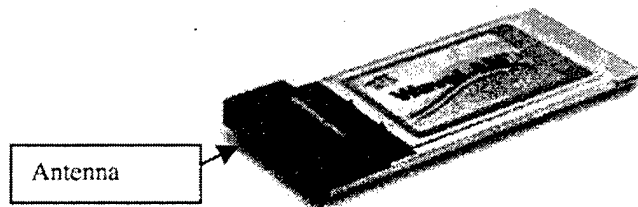


Figure 9. WaveLAN IEEE PC Card - WEP (Silver)

The cards communicate with the WavePoint using DSSS radio signals in the 2400-MHz to 2483.5-MHz band. There are 12 available channels, each of which supports 2 Mbps. Within the radio range, only one transceiver on each channel may be transmitting at a time using CDMA/CD protocols, making an 802.11-based LAN perform

similarly to shared Ethernet. There is no limit to the number of transceivers able to access a single channel.

Each card comes equipped with a fixed internal antenna that sticks out approximately 1-5/16 inches from a PC card slot. Installation of WaveLAN is very simple. Just insert a transceiver into a mobile computer's PC card slot, put the driver floppy in drive A and boot. The WaveLAN Access Point is installed by inserting a transceiver into a PC card slot and running a 10Base-T cable from the Access Point to an Ethernet hub. With a desktop PC, it is necessary first to install the ISA-bus PCMCIA adapter and then install the WaveLAN card.

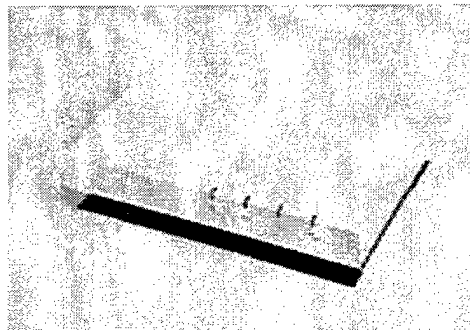


Figure 10. Lucent WavePoint II Access Point

NIC configuration is simple. In most cases, defaults can be used. The user can set the radio-frequency channel and network name, which should be set to enable some minimal security.

According to Lucent, at 2 Mbps, WaveLAN's radius is 1300 feet in an open (line-of-sight) environment and 300 feet in a semi-open space where signals are blocked by walls. The system can automatically drop back to 1 Mbps, extending the range to 1750

feet and 375 feet, respectively. Lucent claims that an antenna will add as much as 15 percent to the radius.

Tests conducted in the laboratory and on the NPS campus achieved connections at distances of about 250 feet (plus or minus 50 feet) through exterior walls (between the Access Point with an indoor antenna and a laptop with a PCMCIA card) and about 740 feet (plus or minus 50 feet) line-of-sight with the indoor antenna (range extender shown in Figure 11) seen from a window.



Figure 11. Lucent WavePoint II Range Extender

Although slow when transferring large files, the WaveLAN is more than adequate for transaction processing, sharing Internet access (it's about the same speed as a T1 line) or for handling smaller files. During the tests, It responds very quickly for Shipnet Software. The difference between wired and wireless is transparent to the users.

The new WaveLAN/IEEE Turbo card shown in Figure 12 offers a high-speed mode that is reliable and provides faster response time for data-intensive network applications. Additionally, the standard mode of the system provides exceptional range and is very robust. It perfectly suites for Shipnet environment.

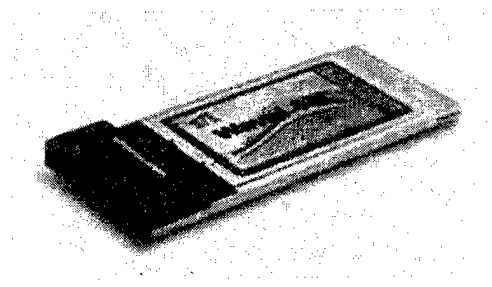


Figure 12. WaveLAN IEEE PC Card - WEP (Bronze)

2. Proxim RangeLAN2

Proxim is one of the worldwide leaders in spread spectrum radio technology, especially in FHSS technology for wireless LANs. Proxim's RangeLAN2 family sets a new standard for spread spectrum radio design. RangeLAN2 is based on frequency hopping spread spectrum technology in the 2.4 - 2.4835 GHz band (like WaveLAN). The unit's average power output is 100 mW. RANGE LAN2's multi-channel approach enables 15 independent, non-interfering wireless LANs to operate within the same physical space, effectively increasing the aggregate capacity of RangeLAN2 networks by 15 times. It uses Proxim version of CSMA/CA protocol optimized for spread spectrum radio.

According to Proxim, at 1.6 Mbps, it supports an effective link up to 700 feet in an open spaces and up to 400 feet in normal office environment. Data rate drops to 800 Kbps for extended range. It provides seamless roaming and is compatible with Wireless

LAN Interoperability Forum (WLIF) products. Both client to server and ad-hoc network architectures can be established.

RangeLAN2 family products consist of access points, LAN adapters (serial, Ethernet, and Token Ring), PCMCIA cards and extension points. The card installation and access points is very similar to that of Lucent products. PCMCIA card needs a Type II slot for portable devices and ISA adapter for desktop computers. These cards connect the portable devices to a wired network via a Proxim access point shown in Figure 13.

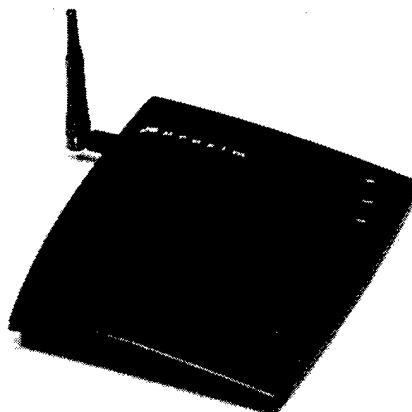


Figure 13. RangeLAN2 7510/20 Ethernet Access Point

There are basically two kinds of antennas for PCMCIA cards. Those are flexible snap-on antenna and dipole antenna shown in Figure 14. Both antennas are designed by Proxim for easy installation, are fully certified by the FCC and ETSI (European Telecommunications Standards Institute) for use with the card and are being approved for use throughout the world. The unique antenna connector approach allows Proxim to design and offer additional antennas in the future that are fully compatible with the current RangeLAN2 PC card products.

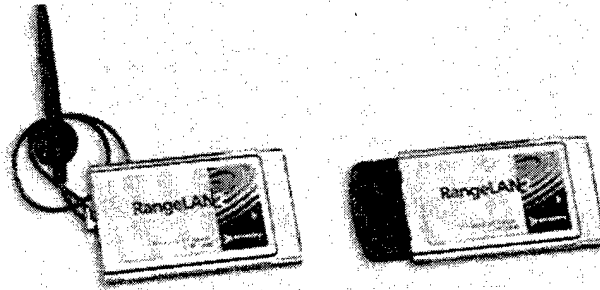


Figure 14. Rangelan2 Dipole and Flexible Snap-On Antenna

C. PORTABLE COMPUTER PRODUCTS

Finding the right portable computer for Shipnet can be a difficult task. There are a lot of different models to choose from and the technology changes so fast that choosing the best solution is important to design long lasting systems. There are some features which should be consider while choosing the portable computers. First, is computing speed and power. Product's microprocessor, memory capacity and overall hardware design including dimensions are important factors to be considered.

Second, is the operating system and supporting software products. The quality of those will affect the performance of the Shipnet's hardware and software. For example, a good handwriting recognition system will improve the utilization of the product. Another, may be the specific connections supported by the mobile unit. For WLAN products mentioned earlier, a Type II PCMCIA card slot is essential for a wireless connection. Some of smaller products, especially PDAs, does not support this vital property, although they have a lot of useful characteristics. Having convenient

connectors to external keyboard, mouse, headset, microphone, monitor, printer and serial devices increases the flexibility of the same device used for another task such as online education, and entertainment.

One of the most important features which should be considered is battery life and power consumption. Increased performance is critical to mobile computing. But if battery life is sacrificed for the sake of performance, there's little gain in productivity or convenience. It's frustrating to have a battery go dead in the middle of a serious situation or after adding the finishing touches to a report.

After all analysis, the final bottleneck for the implementations is usually the price of systems and parts. Currently, the price of the portable computing devices goes from \$99 dollars to a few thousands dollars covering the range from simple PDAs to powerful laptop computers. As it is mentioned earlier, the market is growing rapidly. So the prices are going down.

1. Amity VP

Amity VP is a compact and lightweight portable computer, which combines outstanding portability with high performance. It is shown in Figure 15. It combines the benefits of full Windows 95 and 98 functionality with the right size and weight for the true mobile users. Even though weighing only 2.2 lbs (1 Kg), the Amity VP includes a high performance AMD, Am5x86 processor. Which operates at 133 MHz with up to 48 MB of memory and an 810 MB hard disk drive. The high contrast 7.5 color VGA (Video Graphics Array) "Diamond Fine Passive" LCD (Liquid Crystal Display) screen

displaying 16-bit color and integrated Wacom digitizer enables easy and convenient viewing and control.

A built-in keypad allows fast input of numerical values. Although it is written in the publications that the Handwriter Recognition System (HRS) software from Communication Intelligence Corp. (CIC) provides reliable handwritten input to most applications. Among the portable computers presented in this thesis, it has the least effective HRS software. SoundBlaster Pro compatible hardware enables effective use of sound in any mobile application through the use of the integrated microphone and speaker. Convenient connectors allow easy access to external keyboard, mouse, headset, microphone, monitor, printer and serial devices. It has two Type II PC card slots accepting many types of PCMCIA cards and the optional floppy disk drive.

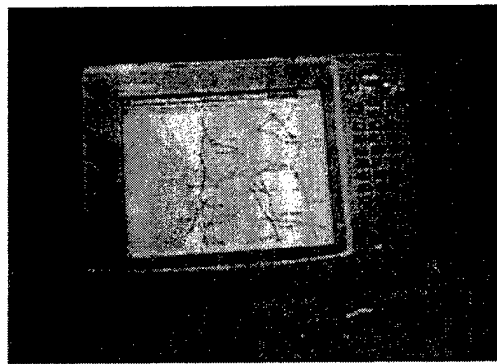


Figure 15. Amity VP Portable Computer

They're two Lithium Ion battery slots. They can be "hot swapped" while the computer is still operating, thus allowing continuous use throughout the work day. A

special charging circuit allows the batteries to be charged anytime the AC adapter is plugged into the Amity VP even while in use [Ref. 20]. According to Mitsubishi, its batteries last up to three hours. However, tests showed that it consumes a battery approximately in half an hour while heavily used. Although its batteries can be hot swapped, because of its higher power consumption a constant flow of full batteries will be needed. In damage control environment, this may cause serious problems. Also it is almost impossible using the device while it is connected to an adapter. Its boot up time is also considerably long compared to that of PDAs and CE devices. Another disadvantages of Amity VP is its size. When compared to other hand held pen tablet computers it is one of the biggest.

The biggest advantage of Amity VP is to run Win 95, which provides more computing power than Win CE Devices. Moreover, it supports JVM for Shipnet software running applets. There are plenty of drivers and commercial software products supporting its functionality.

There are new versions of Amity hand held computer having better features by using different microprocessors such as Amity CP with Pentium II 166 and more advanced hardware. Those products may eliminate Amity VP's current disadvantages.

2. Cassiopedia PA-2400

It is Casio's robust handheld commercial computer using the Microsoft Windows CE 2.0 Operating System. It originally developed to provide a robust mobile solution for data or order entry in commercial market.

The pen-based PA-2400 shown in Figure 16 runs on a Hitachi SH-3 Super-H RISC Processor 80 MHz processor, with 16 Mb of ROM and 8 Mb of RAM. There is no hardware keyboard - all data is entered via the software keyboard or whatever screen-based method is required by application. It has a soft keyboard that comes up on screen. The hard cover folds back flat against the bottom of the unit to make it easy to use. It comes with applications including voice and handwriting recognition. To help with mobile applications, the PA-2400 has options for a barcode reader, modem and printer connection, digital camera, memory cards, and battery packs [Ref. 21].

It is powered by a 3.6 volt Lithium Ion battery. Although PA-2400 has a long battery life, it is observed in the tests that PCMCIA card drains the battery power faster than normal conditions. The same effect was shown for other portable devices as well. But instant boot up procedure of PA-2400 makes frequent ONs -OFFs very handy to save power.

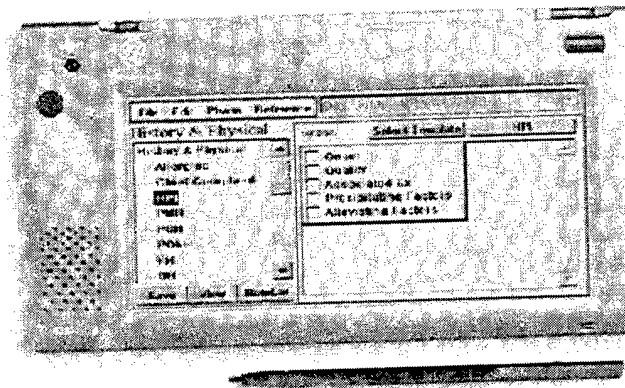


Figure 16. Cassiopedia PA-2400

It has one Type II PCMCIA slot, one Type I CompactFlash™ slot, and one Infrared port (115.2 KPS). Its large back-lit screen, light weight, and powerful features make it ideal for Shipnet. It is small enough for users to carry everywhere allowing for uninterrupted use.

Its build-in features helps implementation of prototype Shipnet software. Applets and servlets using pull down menus and other GUI elements can minimize the number of input errors by users. It reduces the dependency on the quality of users' handwriting. Moreover, it has better handwriting recognition system than Amity has.

PA-2400 has a large screen with larger pixel size, which provides a clear and readable screen. This feature has a great impact on HTML pages posted on a screen by servlets seen more clearly.

Currently, PA-2400 doesn't provide a JVM for running applets. Only servlets can be used for prototype Shipnet software. There are some studies about a JVM running in CE operating system. In the very near future, CE will provide full Java support. Since PA-2400 uses standard Microsoft Windows CE 2.0 operating system, upgrading the device for future versions of OS is very easy. This prevents the handheld units become obsolete quickly.

3. HPW-600ET

HPW-600ET employs the Hitachi SuperH SH-4 RISC processor operating at 128 MHz. It is the lightest PC in its class. It has easy-to-view 7.5", 65536-color, backlighting STN LCD screen that is super-bright and touch resistive with full window functionality. It also offers a built-in 56 Kbps V.90 modem, and 32 MB ROM, 16 MB RAM (max 32

MB). It supports fairly more output connections than many handheld devices. Those are one serial port, one infrared port, one Type II PC card slot, one Type II CompactFlash™ slot, one USB port, Microphone jack , Speaker jack , Headphone jack, and VGA adapter.

The version 3.0 of Windows CE has recently been released. Hitachi didn't waste time to utilize this new OS in its new product, HPW-600ET is shown in Figure 17. It is a combination of handheld and pen-based PC with a tablet design. It has no keyboard set but provides a virtual one. However, its handwriting recognition software is the best one among the devices tested for Shipnet project.

The light-weight design is perfect for mobile users. In addition to the superior lightness, graphics capabilities and speed for its class, the enhancements provided in the tablet-style and new software offer a good solution for Shipnet.



Figure 17. Hitachi's HPW-600ET Tablet

Its big screen is very suitable for displaying whole applet or servlet page at once. Since there will be less scroll up-down or left-right, interaction with the software will be easier and faster. Especially in damage control application, this may be a crucial factor. Also big screen gives the user lots of room to write and/or draw and to place the virtual keyboard to a location where it does not obstruct the windows.

HPW-600ET offers a long-battery life of approximately nine hours. Although PCMCIA card reduces the battery life, it is still better than Amity and PA-2400. In addition, power-down modes of the SH4 microprocessor adds effective power conservation features. Instant On/Instant Off capability also helps to save power. Like PA-2400, currently HPW-600ET does not support Java Virtual Machine. So the same constraints are valid for HPW-600ET as well.

4. VIA II (FLEX) Wearable Computer

It is the world's first and only flexible wearable PC powered by the highly integrated MediaGX processor (available at 180, 200 and 233 MHz) from Cyrix, shown in Figure 18. Via's unique flexible belt-like design allows the unit to be carried comfortably around a user's waist. Wearing the computer at least 6-12 hours per day would be comfortable and not become an obstacle.

It weighs only 22 oz and has 3.2 GB hard drive, 64 MB of DRAM and two PC card slots. The entire unit fits into a pouch which wraps around the user's waist using Via's patented flexible circuitry for maximum comfort and wearability. The system is capable of wireless modem connectivity, including Wide Area Network (WAN), LAN

and cell phone (using two Type II PC card slots), and runs on lithium-ion hot-swappable batteries for up to six hours of run time per battery.

It has a color flat panel touch screen carried in a holster attached to belt. In addition to pen, it can be used through voice (using a headset and multi-lingual voice recognition software). This display also contains buttons for power on/off and mouse functions. The display is available in standard 6.5 inch and 8.4 inch sunlight-readable versions. It is dust and water resistant.

By using a separate docking station, Via II can support small VGA (SVGA) video for desktop display. It also has mouse and keyboard ports, microphone and speaker jacks, a RS-232 serial port, a universal serial bus (USB) port, and two battery recharge slots, plus optional Ethernet connection, CD ROM and external hard drive.

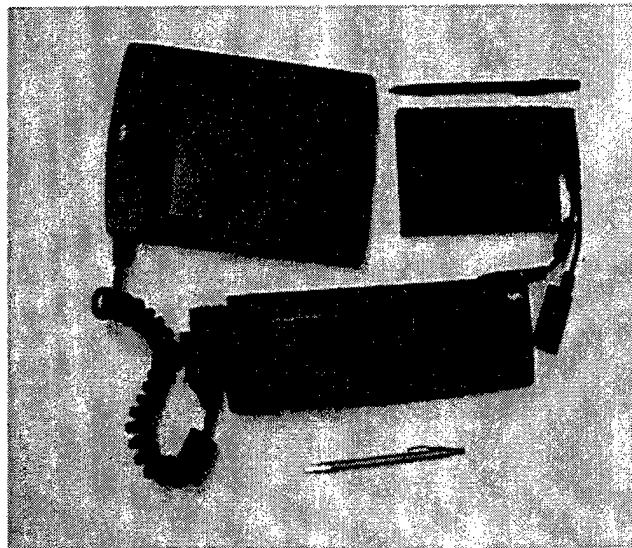


Figure 18. VIA II Wearable Computer (Flex)

One of the most useful features for Shipnet is hands-free mobile computing. Ship personnel in a chaotic time and mission critical environment need a system that is a benefit, not a burden. Another powerful feature is that it can run Windows 95 and 98 operating system. So it supports a JVM which lets applet to run. Also it has some good power management capabilities.

D. SHIPBOARD TESTING

Onboard testing was conducted on the USS HARRY S. TRUMAN. Wireless communication devices, mobile computer equipment and prototype Shipnet software were examined during this testing. Without using TRUMAN's wired LAN, a private network is created by using a laptop as a server. Four different setups were established to test hardware components. By using a file transfer protocol (FTP) program, a large file (5 MB) was transferred from the server to the mobile client and then from the mobile client to the server.

1. Test One

For this phase of the testing, one access point was positioned at the starboard side of the hangar bay. Two-way file transfers were made between the server and one mobile client positioned at different locations relative to the access point by using FTP. By using the average transfer time provided by the FTP server, the data was determined and verified by taking the file size and evaluating it with the total transfer time. The results

provided from these measurements are presented in Figure 19. The locations of the client at the time of measurement are pointed with the arrows.

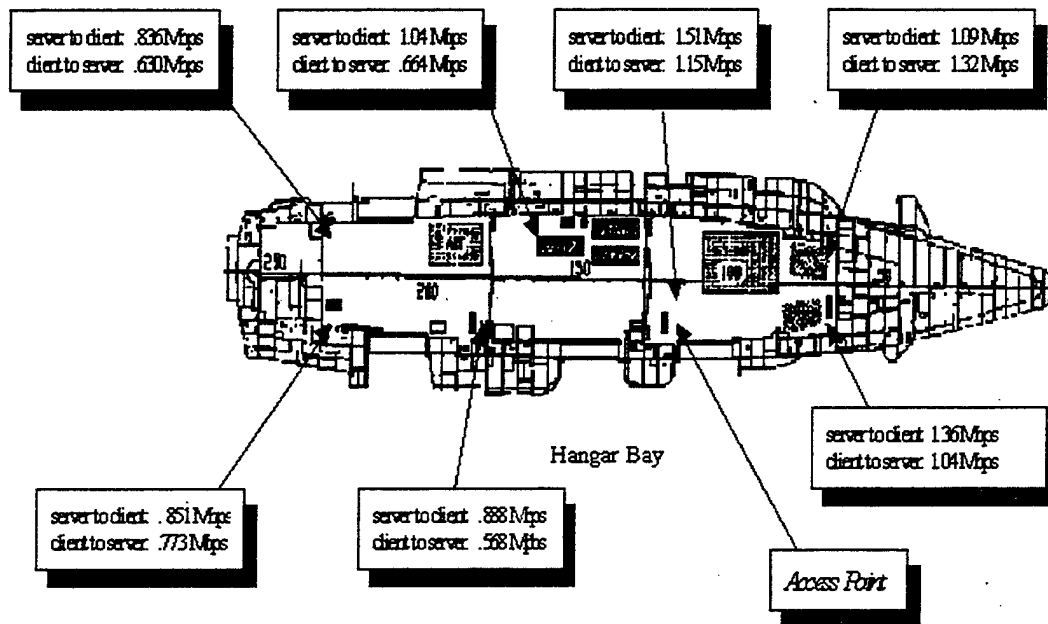


Figure 19. Test One Result

2. Test Two

The access point was stationed at the same location. But two mobile client were used simultaneously to take measurements from the same remote location. The aim was to determine any degradation of transfer rate and throughput in a multi client environment. The results are shown in Figure 20. The term "same direction" in the graph below indicates that the computers were both conducting either simultaneous uploads or downloads. The term "Reverse Direction" implies that while one computer was uploading,

the other was downloading [Ref. 14]. All measurements are in Kbps. The Via II (flex) wearable and amity hand held computers were utilized during the testing.

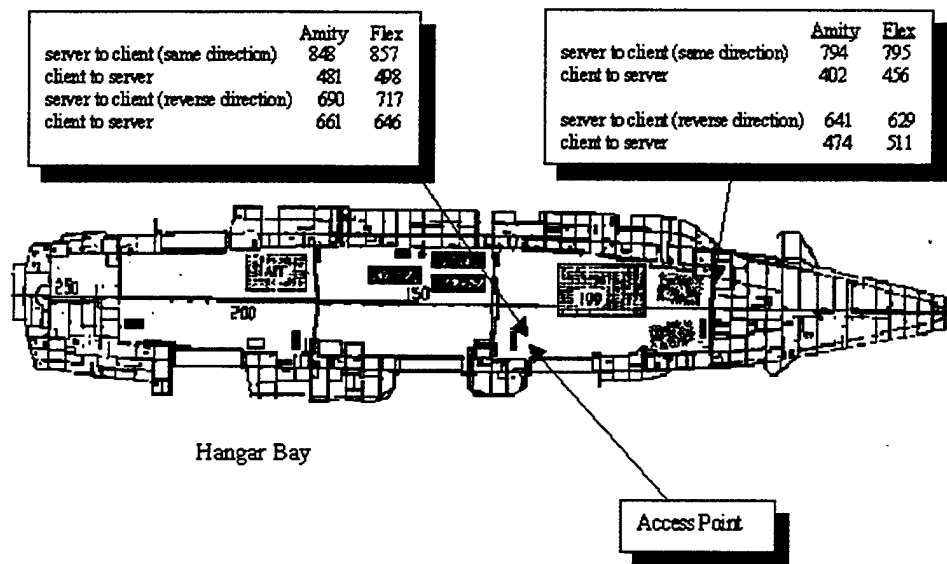


Figure 20. Test Two Result

3. Test Three

In this phase of the test, we had one access point still positioned at the same location and three mobile clients. Via II, Amity and Xybernought's wearable computers were used to send and receive a file simultaneously as mobile client hardware. As we observed in previous two test results, the data rate from server to client was still greater than that of client to server. When the units moved further from the server, the degradation increased, especially while the other two clients were transferring. The tests were conducted in both server to client and client to server mode and configured to determine at the various

ranges the effects of uploading while the other two units were downloading. The test result can be seen from Figure 21. The readings are in Kbps.

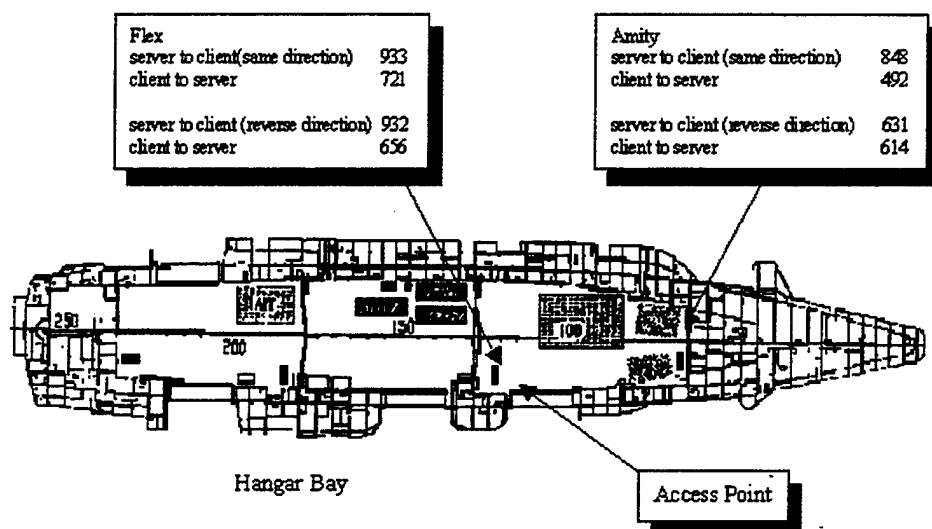


Figure 21. Test Three Result

4. Test Four

This test was planned to see the roaming capability of the wireless network products namely Lucent WaveLAN2 access points and PCMCIA cards. Two access points and one mobile client were used for this test. The access points were located sixty feet apart from each other. Then FTP transfer was initiated while client was close to one of the access points. The client began to move away from its starting point toward the second access point until the client shifted. Almost no degradation was noticed during the shift.

V. PROTOTYPE SHIPNET SOFTWARE APPLICATIONS

A. GENERAL

To take the full advantage of COTS products and to fulfill the unique needs of Navy ships, a prototype software is developed to demonstrate the potential and effectiveness of hardware devices utilized in the proposed WLAN architecture. Any application developed has to be effective, inexpensive and flexible. A Java based intranet solution is proposed in [Ref. 14]. The simplicity and economy of web based software combined with the power and functionality of pen based computers creates a dynamic and effective architecture.

A client-server model, operating in applet-type architecture, creates an application which can operate on PDA hardware. Also Java servlets can be used to provide another solution to the same architecture with a server side computational intensive approach. In the Java applet case, the client side is more computationally intensive. So hardware devices used for the applet side need more CPU power, speed and bigger memory to run efficiently. Moreover, the type of browser used for applet is important. Browsers may show slightly different results for the same application. The two most common browsers, Netscape and Microsoft Internet Explorer, were tested. Small differences of the GUI interface results were observed. Although these differences did not cause any problem for the Shipnet software, it is important to note that there can be small difference in interpreting Java byte codes for different JVMs and browsers.

For servlet applications, client side devices need comparably less computational power and resources. The minimum need for a servlet to run at the client side is to

support HTML. So smaller and slower device can be used for client devices, as long as it has a PCMCIA card slot and an HTML browser. This feature reduces the size, complexity and price of the hand-held units. A light HTML browser can support these servlets. Also supporting a JVM is no longer a necessity. However, having a JVM extends the type of Java based software applications which can be used for some other purposes, such as online instruction or entertainment applications. For this purpose, different kinds of devices can be utilized for different uses.

As mentioned in Chapter 3, servlets are running at the server side of the system like a daemon. They are always ready for a potential request, unless the server is down. When a message from an HTML page activates a servlet, the server runs a program and waits for a new request. The server then posts the result to the browser as an HTML page. HTML's GUI capabilities aren't as good as applet's. This is one of the differences that can be seen from the screen shots given in the figures throughout this chapter. One of the other major differences which can not be seen from the figures is that the degree of interactivity a servlet provides is small when used only with HTML. But by utilizing applet-servlet communication, higher degree of interactivity can be supported.

In this chapter, the applet and the servlet version of the damage control client are provided for comparison. Effects of both of them on Damage Control application will be discussed in detail.

B. GRAPHICAL USER INTERFACE (GUI) POLICIES

The client applet and servlet modules are designed to provide a fast and effective connection with a database. Due to the critical nature of damage control information, robustness and reliability are the most important issues in the development process. Since the application is planned to be used with wearable and pen-based processors connected to a wireless LAN, the GUI should be as simple as possible. Tests showed that pull down menus, buttons, check and radio boxes increase the response time of the user by reducing the input time of the information and complexity of the input device such as pen, virtual keyboard, and voice recognition.

Although voice recognition systems provide a hands-free working environment, they currently are not as good as we need in our application. They are highly prone to errors in noisy environments and depend on the pronunciation of the user. Each user needs to be trained to operate the system. Moreover, the recognition system itself has to be trained for each user's speaking features for reducing these errors. It needs some extra hardware such as microphone, headset, audio adapter to operate, this can be unsuitable for some users working in demanding environments. Navigation on the GUI using voice recognition is not as smooth as the one with pen based systems.

With pen-based computers, there are two choices to input data. The first one is a virtual keyboard. The other is a hand writing recognition system. Writing with a virtual keyboard is slower than doing the same thing with a good hand writing recognition system. Among the hardware devices mentioned previously, Hitachi's CE device has the best hand writing capability. Anyone can almost write with his normal writing speed after small period of time. Casio's Casiopedia CE device and Mitsubishi's Amity have the

second and third places, respectively. Another disadvantage for a virtual keyboard is occupying the screen. Since the screens are already small for hand-held computers, the user needs to drag the keyboard window back and forth while interacting with GUI. On the other hand, using a bad hand writing recognition system causes a lot of errors which degrade the simplicity and fast usability features of the software.

As mentioned above the effective usage of text fields and text areas depends highly on input devices. To reduce input time and provide easy interface, the number and content of these fields are minimized through the development of the software. In this thesis, it was observed that activating the fields on the screen with a pen is much more easier and faster than the same process via a mouse on a desktop computer. Future development of the code and the hardware devices will clearly make the interface much more user friendly and powerful.

C. DAMAGE CONTROL CLIENT APPLET

For the damage control, the first goal was to create an application which can support fast and reliable data for a command and control systems. By using a WLAN and Java together, this feature was provided inherently. The latest portable devices can support almost real-time data by utilizing Java's robust connection properties for networks and databases. An Intranet/SQL style architecture proposal by Rothenhaus [Ref. 14] lies under the database connection to this application.

The second goal was to provide an easy interface for DC's relatively complex

structure. The DC environment is one of the most demanding and critical situations faced by the submarine crew. So an extra care was spend to find a good representation of the DC system. To create pages used in electronic DC form, a DC manual of the attack submarines and current paper DC forms were examined. Also some Navy officers proposed a draft and checked final version of the finished product.

The form of DC client consists of five different tabs. Each of them provides unique properties for the DC client (fire) applet. These tabs can be seen in Figures 22 through 26. To reduce the execution time of the applet, each DC module's properties were designed to place into the same applet as selected by tabs. Other prospective modules such as flooding, major steam leak can be also added to the same applet as different tabs. But this increases the file size which may cause some problems for the application. There is a trade-of between running big files into small hand held computers and having a smooth working smaller applications. The former may cause slower response, loading and initializing errors. Writing other modules as different applets can provide better usability and responsiveness. The code is provided in Appendix A.

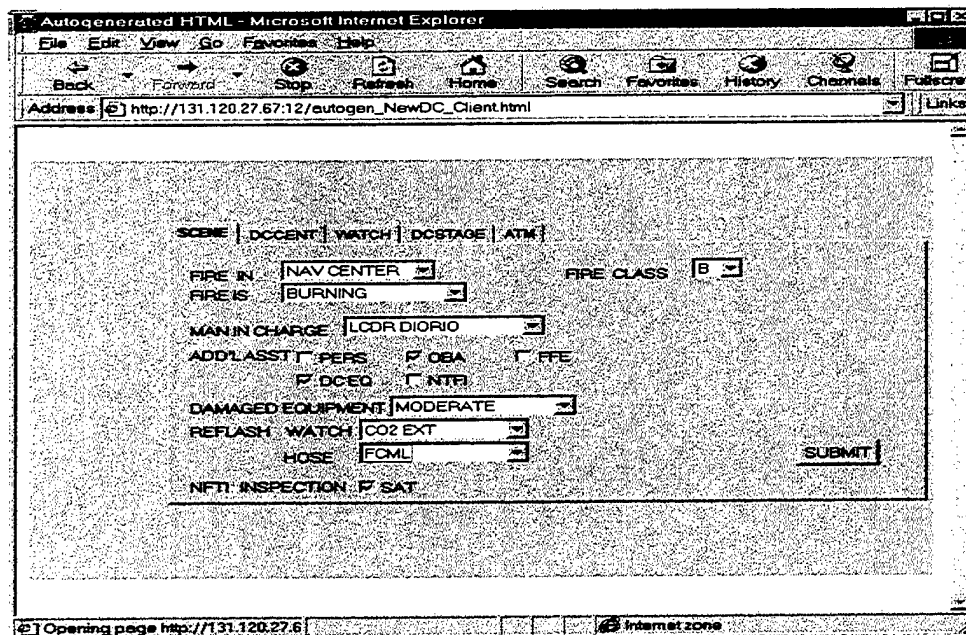


Figure 22. DC Client (Fire) Applet Scene Page

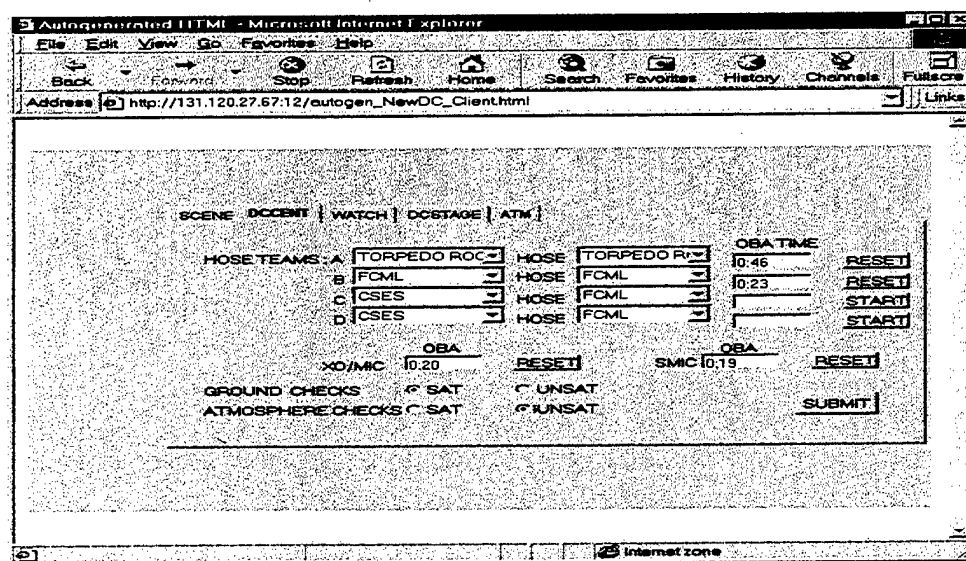


Figure 23. DC Client (Fire) Applet Central Page

Autogenerated HTML - Microsoft Internet Explorer

File Edit View Go Favorites Help

Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen

Address http://131.120.27.67:12/autogen_NewDC_Client.html Links

SCENE | OCCENT | WATCH | DCSTAGE | ATM |

REPORTING STATION: NAV CENTER

COMPARTMENT RIG:

☐ FIRE \ GEN EMERG

☐ SNORKEL \ VENT

☐ RFFE

SUBMIT

Internet zone

Figure 24. DC Client (Fire) Wachtstander Page

Autogenerated HTML - Microsoft Internet Explorer

File Edit View Go Favorites Help

Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen

Address http://131.120.27.67:12/autogen_NewDC_Client.html Links

SCENE | OCCENT | WATCH | DCSTAGE | ATM |

MAN IN CHARGE: LT DWYER

HOSE TEAM	OBA TIME	
HOSE TEAM A	0:17	RESET
HOSE TEAM B	0:16	RESET
HOSE TEAM C		START
HOSE TEAM D		START
XO / MIC	0:9	RESET
SNIC	0:8	RESET

SUBMIT

Internet zone

Figure 25. DC Client (Fire) Staging Page

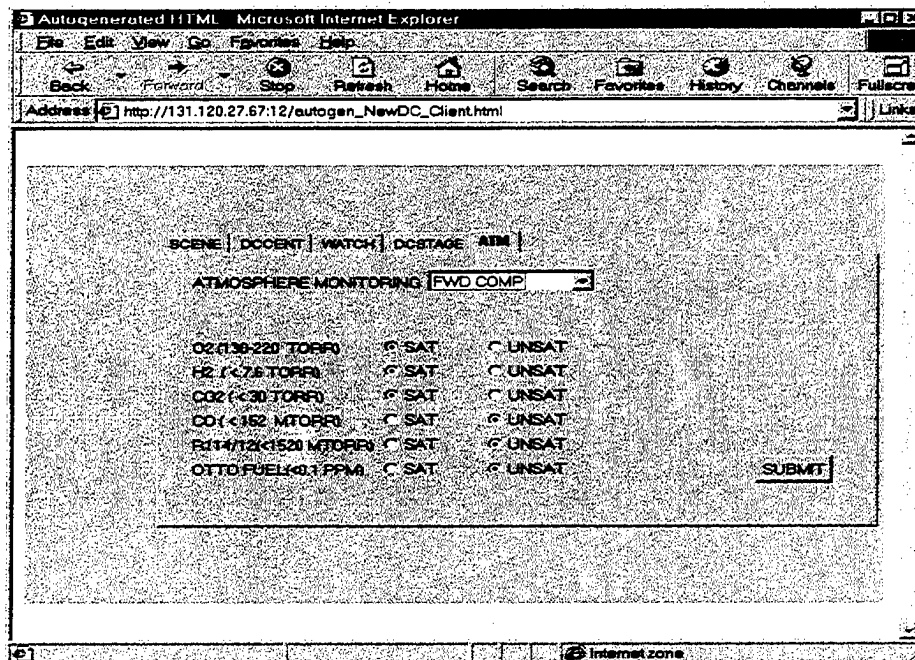


Figure 26. DC Client (Fire) Atmosphere Control Page

D. DAMAGE CONTROL CLIENT SERVLET

The applet and the servlet version of the Damage Control client supports the same functionality for the database. They both provide almost real time and reliable data flow for the system. Some of the architectural differences are given in Chapter 3 and at the beginning of this chapter.

Because of the graphical user interface capacity of the HTML, some changes are made to sustain a simple GUI for the servlet user. HTML does not support the tab style presentation of the client application. So each DC page has to be written on different

HTML page. To orchestrate the sub-pages belonging to the same module, a menu page seen in Figure 27 was inserted into the servlet package of the DC client module.

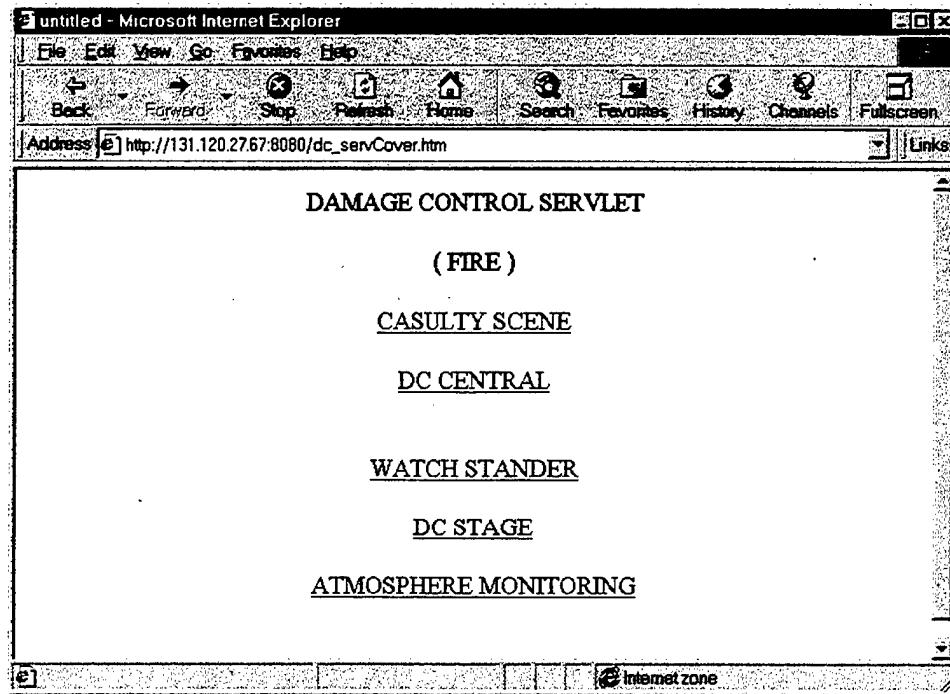


Figure 27. DC Client Servlet Menu Page

Since the content of these pages are similar to the pages in the applet client, only Casualty Scene page is given in Figure 28 to show the visual differences between applet and a servlet.

Behind the scene, servlet module operates in the same manner that the DC client applet does. The only exception are the SQL queries and updates in the server module do not use any non-standard Java class namely JDBC01 provided in Appendix B. Servlet directly connects to the database by utilizing the Java sql package.

When a page's hyper link is activated from the menu page, corresponding servlet's certain methods are invoked by the server in the process of handling requests. To be exact, each time the server dispatches a request to a servlet, it invokes the servlet's `service()` method. This method handles the requests and responses. So it posts the HTML page on the screen of the user device. After first initialization of the page each request from the page is handled with a similar fashion. The code for the DC client servlet is provided in Appendix C. In this thesis Java Web Server 2.0 is used. But there are a lot of servers utilizing Servlet API. The number of the server having servlets is increasing. All major web servers are expected to support servlets shortly [Ref. 16].

For providing a feedback messages to the user, the DC client servlet posts a new HTML page for the user device. This page consists of the data currently written to the database. The system and the user errors are also written on this page. Although this is totally a optional choice, it has some advantages. One of them is to prevent a user entering wrong data into the database which may cause some serious consequences, due to critical nature of the application. Another advantages is in the event of a system failure, database may not have been modified and the user may not know whether the data reached the database or not. By using this kind of feedback page, user knows the real status of the database and can take further steps without hesitating. A simple view of the feedback page is given in Figure 29.

By using the browser's *back* and *forward* button, the user can navigate easily inside the application.

untitled - Microsoft Internet Explorer

File Edit View Go Favorites Help

Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen

Address http://131.120.27.67:8080/DC_ClientServlet.htm

FIRE IN CSES

FIRE CLASS A

FIRE IS BURNING

MAN IN CHARGE LCDR DIORIO

ADD'L ASST ☐ PERS ☐ OBA ☐ FFE

☐ DC EQ ☐ NTFI

DAMAGED EQUIPMENT NONE

REFLASH WACTH FIRE HOSE

HOSE FCML

NTFI INSPECTION ☐ SAT

Submit

Internet zone

Figure 28. DC Client Servlet Casualty Scene Page

DC Report - Microsoft Internet Explorer

File Edit View Go Favorites Help

Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen

Address http://131.120.27.67:8080/servlet/DC_ClientServlet

DC Updated: Press Back button to return

FIRE_IN=CSES
 FIRE_CLASS=A
 FIRE_IS=BURNING
 MAN_CHAR=LCDR DIORIO
 PERS=yes
 OBA=no
 DC_EQ=no
 NTFI=yes
 FFE=no
 SAT=yes
 DAM_EQ=NONE
 WATCH=FIRE HOSE
 HOSE=FCML

Internet zone

Figure 29. DC Client Servlet Feedback Page

E. LOG TAKING CLIENT APPLET

The Log Taking module is the second application of the Shipnet implementation which is investigated in the scope of this thesis. This module provides a fast means of data flow from watch standers to the high level command structures. In a submarine environment, instrument readings are taken each hour and written onto a log sheet. Out-of-range readings are circled in red and reported. Normally supervisors review these logs every few hours to keep track of the current condition of the ship. By using Shipnet, this kind of continuous data flow can be saved in databases more efficiently and quickly. Moreover, supervisors can reach the data whenever and wherever it is needed. Robust and dependable warning systems can be implemented by processing the electronic data. Another advantages of this application is to reduce the amount of resources such as space, man power spent for saving and retrieving old records.

The log taking application was modeled after the Auxiliary Electrician Forward (AEF) equipment logs. The logs basically consist of two forms: one for atmosphere monitoring and one for auxiliary electrician use. An extra *Remarks* form is provided to comment about the current situation. All forms are designed to provide the previous six hours of logs and allow for the entry of the current equipment status. This six hour history field is added to the application to view the condition of the devices over a period of time at the first glance.

There are some built-in security features embedded into the application for both the applet and the servlet. These features enhance the control of the watchstanders and the integrity of the data. One of these is that only the current time period's entry is accepted. If anybody tries to enter some data before or after the designated time period,

database will not accept the entry. It enforces punctuality for each duty stations. The previously submitted log entries can not be changed by a later entry. These feature protects the integrity of the data and may be more important for some other applications too.

The form of Log Taking client consists of four different tabs. Each of them provides unique properties for the Log Taking client (AEF) applet. This tabs can be seen in Figures 30 through 32 and the code for the applet is provided in Appendix D.

Autogenerated HTML - Microsoft Internet Explorer

File Edit View Go Favorites Help

Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail

Address http://131.120.27.67:12/autogen_yedekLog.html Links

AEF Logs | Atmosphere Control Logs | Remarks

480 Cycle Sets | Detectors & Indicators

NO. 1						NO. 2					
Time Perio	DCAm	ACAm	Freque	Volts	Gnd D	DCAm	ACAm	Freque	Volts	Gnd D	
0:00	15	20	240	110	0	15	20	243	112	0.1	
1:00	15	21	242	110	0	14	21	243	110	0.1	
2:00	16	20	241	115	0.2	12	21	245	112	0.2	
3:00	15	23	241	113	0.3	13	23	243	113	0.3	
4:00	14	22	240	111	0	14	20	244	114	0.4	
5:00	15	22	245	108	0	15	18	245	115	0.5	
06:00	15	20	240	110	0.2	15	20	245	110	0	

Retrieve Submit

Applet started Internet zone

Figure 30. Log Taking Client Applet

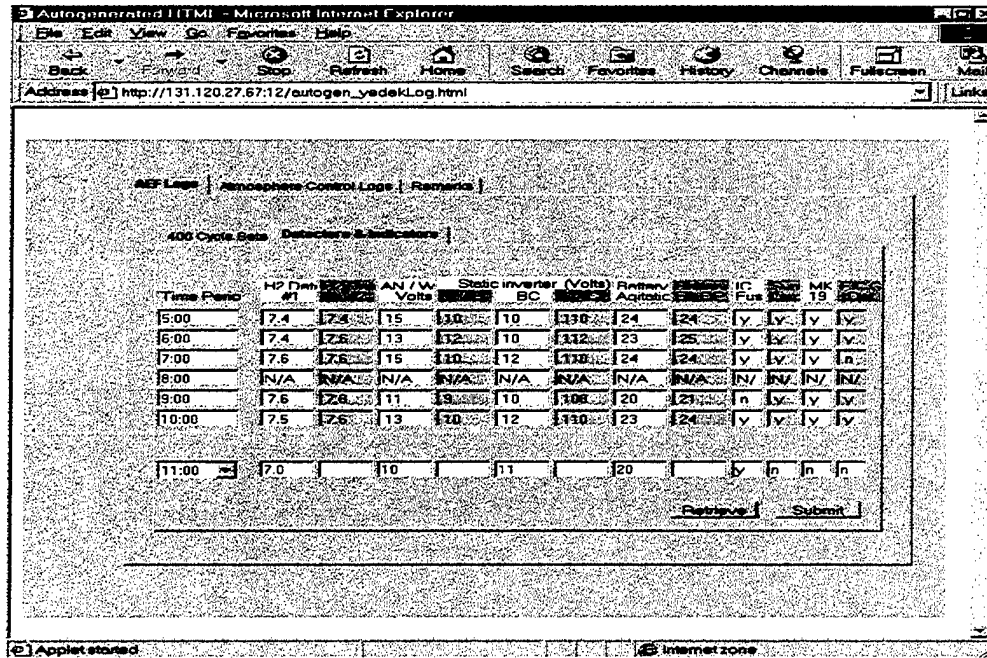


Figure 31. Log Taking Detectors Page

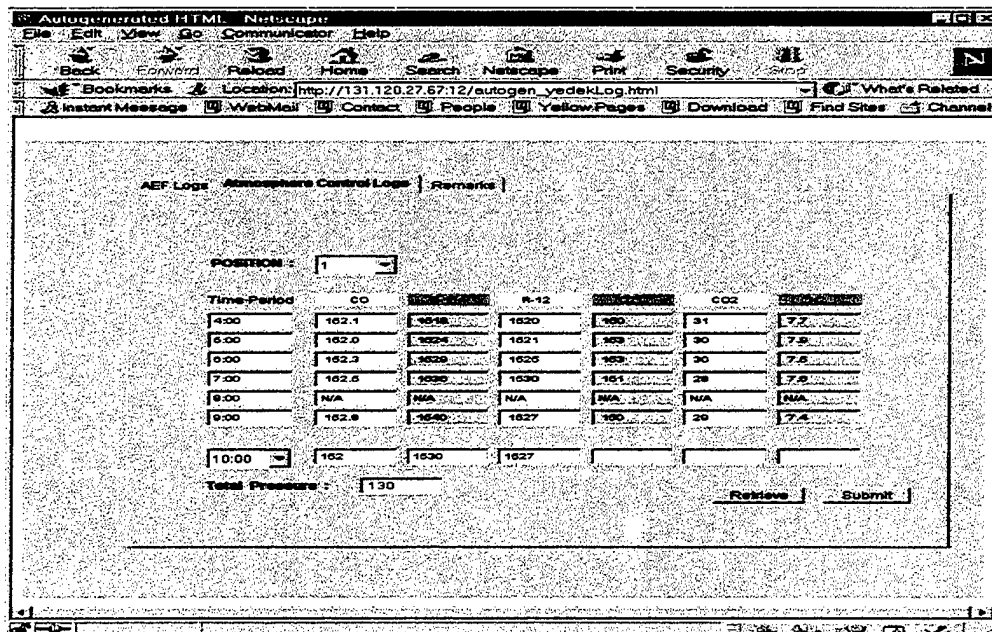


Figure 32. Log Taking Atmosphere Control Page

F. LOG TAKING CLIENT SERVLET

Similar to the DC servlet, a menu page with hyper links is utilized for Log Taking to coordinate the different pages, see Figure 33.

Although both DC and Log Taking servlets architecturally have the similar properties, Log Taking servlet has a different kind of implementation approach than the DC Client. The code is provided in Appendix E. This difference is caused by the nature of both applications. In the DC application, servlets only updates the database for each casualty scene. It does not need any data to pass to the user. So it is very straight forward. However, in Log Taking application user needs some data from the database to understand the previous situation.

The main HTML page of the each form has a *MODE* pull down menu, shown in Figure 34, which changes the function of that particular form. The same form can be used to retrieve the previous entries or to submit current time period's data. When *Archive* mode is used, a request is sent to the server to initiate a particular servlet to get the data and post it to the user's computer. Figure 35 shows the table posted to the screen with retrieved data. In *Record* mode it works as it was explained in DC Client servlet and only modifies the database. The error messages and warnings are posted to the screen for feedback shown in Figure 36.

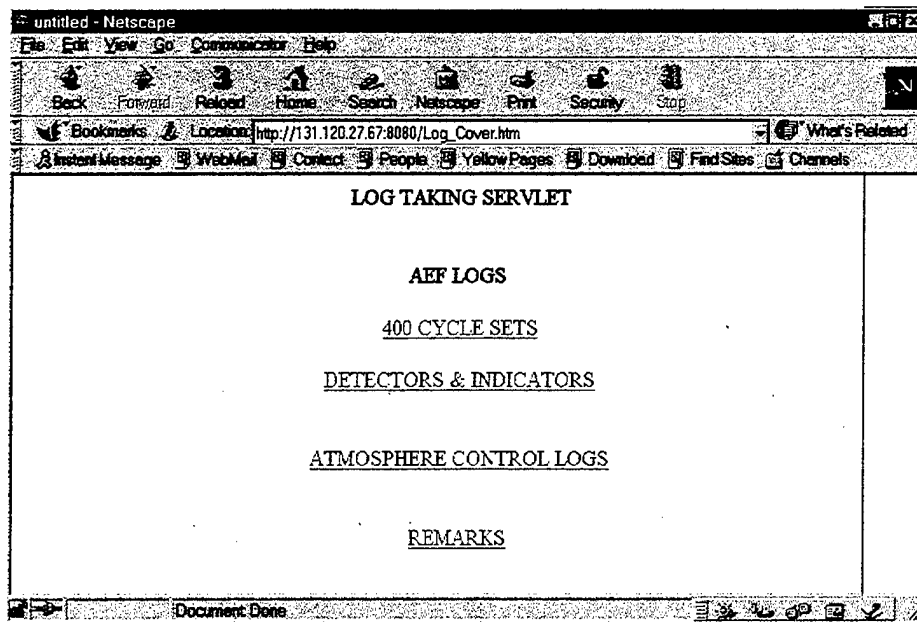


Figure 33. Log Taking Menu Page

untitled - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Stop

Bookmarks Location: http://131.120.27.67:8080/Log_cycleSet.htm What's Related

Instant Message WebMail Contact People Yellow Pages Download Find Sites Channels

400 CYCLE SETS LOG

TIME 00:00 MODE Archive

NO.1 NO.2

DC AMP.....DC AMP.....

AC AMP.....AC AMP.....

FREQ.....FREQ.....

VOLTS.....VOLTS.....

GRND.....GRND.....

Reset Submit

IF YOU WANT TO SEE PREVIOUS DATA BEFORE 'SUBMIT',
PLEASE CHANGE THE MODE FIELD TO 'ARCHIVE'.

Document Done

Figure 34. Log Taking AEF Log Page

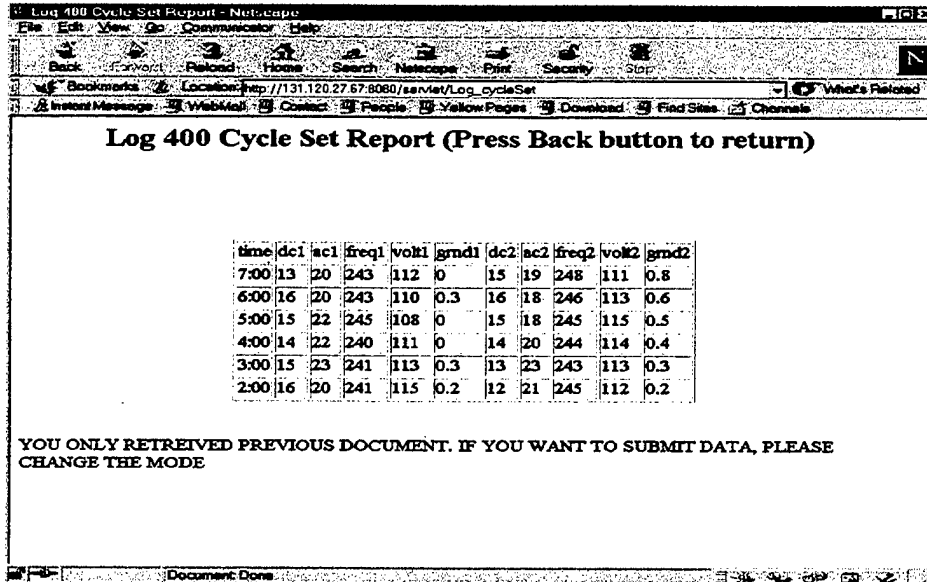


Figure 35. Log Taking Retrieved Data Page

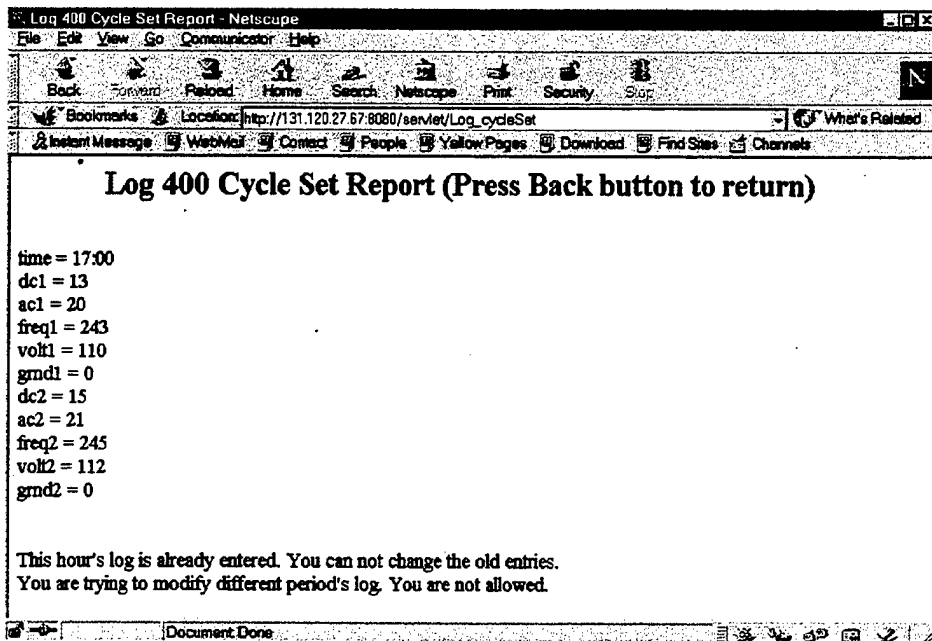


Figure 36. Log Taking Feedback Page

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

COTS wireless LAN and handheld computer products were investigated for shipboard LANs. For WLAN products, the main concern was multi-path-fading effect of the metal hull on radio frequency components. The number of access points required to cover a ship and the number of user machines directly affect the cost and the speed of the WLAN. If the number of the APs goes up, the cost goes up. If the number of the clients served from the same AP goes up, the data rate goes down. Transmission range, throughput, IEEE 802.11 compliance, power consumption, and cost issues were evaluated. The results of the laboratory and shipboard tests conducted onboard the *USS Memphis* (SSN-691) in August 1999 showed that WLAN is feasible for onboard network applications.

To provide the best COTS handheld and wearable computer products for shipboard WLAN environment, Mitsubishi's Amity VP, Casio's Cassiopedia PA-2400, Hitachi's HPW-600ET, and VIA's Flex mobile computers were evaluated for input method, OS, battery life, Java software adaptability, cost and comfort. Although machines with full OS have some advantages like better Java and software support, CE devices have their own advantages like low cost, long battery life, and small dimensions. Pen-based portable devices are useful to utilize for shipboard WLAN applications.

Two different applications were developed for prototype Shipnet software. Damage control and log taking applications were written in Java with a web browser type of interface. The main goals were to create a robust database connectivity and user

friendly interfaces for specific applications. Java's modular, distributed and simple nature helped to achieve those goals. The reporting agent on the client side was written in both an applet and a servlet. Since JVM is supported by almost every web browser, applets are suitable to create web-based software applications for handheld PCs. For CE devices with light browsers which do not support a JVM, Java servlets were used to create applications with the same functionality as applets provide. In this process HTML pages were used as GUI of servlets at the client side devices.

Tests and demonstrations conducted in laboratory and onboard indicated that the prototype Java software is totally compatible with WLAN and mobile computer products and useful. Moreover, Feedback received from crewmen during shipboard testing supported this conclusion.

B. FURTHER RESEARCH

Although the most commercial WLAN and mobile computer products in the market were screened and some of them have been tested for onboard intranet solutions by NPS Wireless Network Research Group, new generation of more powerful products are introduced to the market everyday. So evaluating new products should continue. One of the best examples is that Lucent increased the data rate of its PCMCIA card from average 2 Mbps to 11 Mbps by introducing a new turbo card recently.

Voice recognition systems are developing rapidly as well. In the future, those programs will be mature enough to be utilized in more demanding environments. A further research should continue to integrate voice-input capabilities into Shipnet. The

feedback gained from a Shipnet survey showed that crewmen's first choice is voice input. When integrated efficiently, this product can provide fast and reliable input and hand-free working environment for users.

To take advantage of both applet and servlet Java applications, applet-servlet communication can be established. Better applications can be developed by utilizing suitable features of both. For example, a better GUI can be created by applets. But for background computations, servlet is faster and needs less computational power.

In the scope of this thesis, only Shipnet damage control and log taking modules were created. In prior theses, maintenance, supply inventory and central console modules were developed. However there are still some non-tactical shipboard systems that can be potentially implemented by Shipnet prototype software.

Depending on the computational power of portable devices 3D images can be incorporated with Java. For example, in damage control all casualties and specific important locations such as compartments, hoses can be seen on a screen momentarily by simply submitting a DC client form from a user machine to the database server. This implementation is especially useful for central control applications. It may take the place of status boards.

Two new application areas can be provided for crew by using same infrastructure. Those are online interactive technical support and entertainment. In every situation, help can be "one touch" away from the user. Or one can get email or surf on the Web in his leisure time.

C. CONCLUSION

It is a fact that accurate and timely information is power to win a battle. But to get that information, we have to provide correct tools to those who need it. The correct tool to leverage information technology for afloat Navy is to utilize mobile computing.

In this thesis the combination of wireless LAN technology and mobile computer devices operating Java based software is examined. Although shipboard and submarine environment have unique difficulties to implement WLAN technology, today's network technology and high tech equipment are able to overcome those problems.

The wireless network and portable computer products examined in this research can provide enough capacity to create an effective onboard wireless intranet solution, Shipnet, for non-tactical systems. Moreover, this solution comes with a lot of benefits such as low cost (installation to maintenance), higher flexibility, reliable information flow from down to top and vice versa, quick reaction time, computing power at every level etc.

To meet the needs of both Shipnet and different platforms utilized in Navy's ships and submarines, prototype Java-based software is designed. Java was chosen, because of distributed, platform independent and robust nature of the language. It fits all the different kinds of hardware like a glove. Besides, development and maintenance of the code is very easy when compared to other object-oriented languages like C++ or ADA.

Shipnet like wireless intranet applications are the future of mobile computing for dynamic working environments. But we don't have to wait till tomorrow, because all of the parts are currently available in our hands.

APPENDIX A. DAMAGE CONTROL CLIENT APPLET CODE

```
// DC Client Applet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Applet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Applet can be viewed at:
// http://131.120.27.67:12/dc.html
// Date : 30 Sept. 1999

import java.io.*;
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
import java.util.*;
import java.util.Date;
import java.sql.*;
import java.net.*;
import java.lang.*;

import symantec.itools.awt.TabPanel;
import symantec.itools.awt.shape.VerticalLine;
import symantec.itools.util.Timer;

import symantec.itools.awt.StateCheckBox;

public class dc_Client extends Applet implements Runnable,
ActionListener
{
    public void start()
    {
        super.start();

        try
        {
            theJDBC.openConnection();
        } // end if
        catch (Exception e)
        {
            handleException(e);
        } // end catch
    } // end method start

    public void stop()
    {
        try
        {
            theJDBC.closeConnection();
        }
    }
}
```

```

        watchThread.stop();
    } // end try
    catch (Exception e)
    {
        handleException(e);
    } // end catch
} // end method stop

public void init()
{
    super.init();
    // Take out this line if you don't use symantec.itools.net.RelativeURL
    // or symantec.itools.awt.util.StatusScroller
    symantec.itools.lang.Context.setApplet(this);

    // This code is automatically generated by Visual Cafe when
you add
    // components to the visual environment. It instantiates
and initializes
    // the components. To modify the code, only use code syntax
that matches
    // what Visual Cafe can generate, or Visual Cafe may be
unable to back
    // parse your Java file into its visual environment.
    //{{{INIT_CONTROLS
    setLayout(null);
    setSize(691,441);
    setBackground(new Color(12632256));
    tabPanel1 = new symantec.itools.awt.TabPanel();
    try {
        java.lang.String[] tempString = new
java.lang.String[5];
        tempString[0] = new java.lang.String("SCENE");
        tempString[1] = new java.lang.String("DCCENT");
        tempString[2] = new java.lang.String("WATCH");
        tempString[3] = new java.lang.String("DCSTAGE");
        tempString[4] = new java.lang.String("ATM");
        tabPanel1.setPanelLabels(tempString);
    }
    catch(java.beans.PropertyVetoException e) { }
    try {
        tabPanel1.setCurrentPanelNdx(4);
    }
    catch(java.beans.PropertyVetoException e) { }
    tabPanel1.setBounds(96,60,600,312);
    add(tabPanel1);
    panel1 = new java.awt.Panel();
    panel1.setLayout(null);
    panel1.setVisible(false);
    panel1.setBounds(12,33,576,268);
    panel1.setBackground(new Color(12632256));
    tabPanel1.add(panel1);
    label300 = new java.awt.Label("FIRE CLASS");

```

```

label300 = new java.awt.Label("FIRE  CLASS");
label300.setBounds(300,21,80,20);
panel1.add(label300);
label3 = new java.awt.Label("FIRE  IN");
label3.setBounds(12,21,60,20);
panel1.add(label3);
label7 = new java.awt.Label("FIRE IS ");
label7.setBounds(12,42,60,20);
panel1.add(label7);
label8 = new java.awt.Label("REFLASH    WATCH");
label8.setBounds(12,189,125,12);
panel1.add(label8);
label9 = new java.awt.Label("HOSE");
label9.setBounds(84,212,48,20);
panel1.add(label9);
label10 = new java.awt.Label("NFTI    INSPECTION");
label10.setBounds(12,240,123,24);
panel1.add(label10);
choice2 = new java.awt.Choice();
choice2.addItem("CSES");
choice2.addItem("CONTROL");
choice2.addItem("NAV CENTER");
choice2.addItem("FCML");
choice2.addItem("TORPEDO RCOM");
choice2.addItem("AMR");
choice2.addItem("ERUL");
choice2.addItem("ERML");
choice2.addItem("ERF");
choice2.addItem("TGLO BAY");
choice2.addItem("COND. BAY");
choice2.addItem("MSW BAY");
choice2.addItem("SHAFT ALLEY");
choice200 = new java.awt.Choice();
choice200.addItem("A");
choice200.addItem("B");
choice200.addItem("C");
try {
    choice2.select(0);
    choice200.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice2);
panel1.add(choice200);
choice2.setBounds(84,15,120,40); //40
choice200.setBounds(400,15,40,20);
choice5 = new java.awt.Choice();
choice5.addItem("BURNING");
choice5.addItem("SPREADING");
choice5.addItem("CONTROLLED");
choice5.addItem("OUT");
choice5.addItem("OVERHAULED");
try {
    choice5.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice5);

```

```

choice5.setBounds(84,39,144,40);
choice6 = new java.awt.Choice();
choice6.addItem("FIREHOSE");
choice6.addItem("CO2 EXT");
choice6.addItem("AFFF EXT");
choice6.addItem("PPK EXT");
try {
    choice6.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice6);
choice6.setBounds(144,183,132,40);
choice7 = new java.awt.Choice();
choice7.addItem("FCML");
choice7.addItem("TORPEDO ROOM");
choice7.addItem("AMR");
choice7.addItem("ERUL FWD");
choice7.addItem("ERUL AFT");
choice7.addItem("TGLO BAY");
choice7.addItem("MSW BAY");
try {
    choice7.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice7);
choice7.setBounds(144,207,132,40);
label4 = new java.awt.Label("MAN IN CHARGE");
label4.setBounds(12,80,110,20);
panel1.add(label4);
label5 = new java.awt.Label("ADD'L ASST");
label5.setBounds(12,110,79,12);
panel1.add(label5);
label6 = new java.awt.Label("DAMAGED EQUIPMENT");
label6.setBounds(12,165,152,12);
panel1.add(label6);
choice3 = new java.awt.Choice();
choice3.addItem("LCDR DIORIO");
choice3.addItem("LCDR WOLD");
choice3.addItem("LT MUGGLEWORTH");
choice3.addItem("LT JONES");
choice3.addItem("LT DWYER");
choice3.addItem("LTjg INDELOCATO");
choice3.addItem("ETC SMITH");
choice3.addItem("EMC FOSTER");
choice3.addItem("MMC ALEMAN");
try {
    choice3.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice3);
choice3.setBounds(132,75,156,40);
choice4 = new java.awt.Choice();
choice4.addItem("NONE");
choice4.addItem("MINOR");
choice4.addItem("MODERATE");
choice4.addItem("EXTENSIVE");

```

```

try {
    choice4.select(0);
}
catch (IllegalArgumentException e) { }
panel1.add(choice4);
choice4.setBounds(168,159,144,40);
button13 = new java.awt.Button();
button13.setLabel("SUBMIT");
button13.setBounds(480,207,62,26);
button13.setBackground(new Color(12632256));
button13.addActionListener(this);
panel1.add(button13);
checkbox1 = new java.awt.Checkbox("PERS");
checkbox1.setBounds(96,111,63,15);
panel1.add(checkbox1);
checkbox2 = new java.awt.Checkbox("OBA");
checkbox2.setBounds(180,111,60,15);
panel1.add(checkbox2);
checkbox3 = new java.awt.Checkbox("FFE");
checkbox3.setBounds(264,111,60,15);
panel1.add(checkbox3);
checkbox4 = new java.awt.Checkbox("DC EQ");
checkbox4.setBounds(96,135,60,15);
panel1.add(checkbox4);
checkbox5 = new java.awt.Checkbox("NTFI");
checkbox5.setBounds(180,135,60,15);
panel1.add(checkbox5);
checkbox6 = new java.awt.Checkbox("SAT");
checkbox6.setBounds(144,245,60,15);
panel1.add(checkbox6);
panel2 = new java.awt.Panel();
panel2.setLayout(null);
panel2.setVisible(false);
panel2.setBounds(12,33,576,268);
panel2.setForeground(new Color(0));
panel2.setBackground(new Color(12632256));
tabPanel1.add(panel2);
label11 = new java.awt.Label("HOSE TEAMS : A");
label11.setBounds(24,35,110,12);
panel2.add(label11);
choice8 = new java.awt.Choice();
choice8.addItem("CSES");
choice8.addItem("CONTROL");
choice8.addItem("NAV CENTER");
choice8.addItem("FCML");
choice8.addItem("TORPEDO ROOM");
choice8.addItem("AMR");
choice8.addItem("ERUL");
choice8.addItem("ERML");
choice8.addItem("TGLO BAY");
choice8.addItem("COND. BAY");
choice8.addItem("MSW BAY");
choice8.addItem("SHAFT ALLEY");
try {
    choice8.select(0);
}

```

```

catch (IllegalArgumentException e) { }
panel2.add(choice8);
choice8.setBounds(140,27,118,40);
label12 = new java.awt.Label("HOSE");
label12.setBounds(264,31,38,20);
panel2.add(label12);
choice9 = new java.awt.Choice();
choice9.addItem("FCML");
choice9.addItem("TORPEDO ROOM");
choice9.addItem("AMR");
choice9.addItem("ERUL FWD");
choice9.addItem("ERUL AFT");
choice9.addItem("TGLO BAY");
choice9.addItem("MSW BAY");
try {
    choice9.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice9);
choice9.setBounds(312,27,104,40);
label13 = new java.awt.Label("OBA TIME");
label13.setBounds(432,15,64,20);
panel2.add(label13);
textField1 = new java.awt.TextField();
textField1.setBounds(432,37,60,18);
panel2.add(textField1);
button1 = new java.awt.Button();
button1.setLabel("START");
button1.setBounds(516,38,50,18);
button1.setBackground(new Color(12632256));
panel2.add(button1);
label14 = new java.awt.Label("B");
label14.setBounds(123,54,12,24);
panel2.add(label14);
choice10 = new java.awt.Choice();
choice10.addItem("CSES");
choice10.addItem("CONTROL");
choice10.addItem("NAV CENTER");
choice10.addItem("FCML");
choice10.addItem("TORPEDO ROOM");
choice10.addItem("AMR");
choice10.addItem("ERUL");
choice10.addItem("ERML");
choice10.addItem("TGLO BAY");
choice10.addItem("COND. BAY");
choice10.addItem("MSW BAY");
choice10.addItem("SHAFT ALLEY");
try {
    choice10.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice10);
choice10.setBounds(140,51,118,40);
label15 = new java.awt.Label("HOSE");
label15.setBounds(264,54,38,20);
panel2.add(label15);

```

```

choice11 = new java.awt.Choice();
choice11.addItem("FCML");
choice11.addItem("TORPEDO ROOM");
choice11.addItem("AMR");
choice11.addItem("ERUL FWD");
choice11.addItem("ERUL AFT");
choice11.addItem("TGLO BAY");
choice11.addItem("MSW BAY");
try {
    choice11.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice11);
choice11.setBounds(312,51,104,40);
textField2 = new java.awt.TextField();
textField2.setBounds(432,63,60,18);
panel2.add(textField2);
button2 = new java.awt.Button();
button2.setLabel("START");
button2.setBounds(516,63,50,18);
button2.setBackground(new Color(12632256));
panel2.add(button2);
choice12 = new java.awt.Choice();
choice12.addItem("CSES");
choice12.addItem("CONTROL");
choice12.addItem("NAV CENTER");
choice12.addItem("FCML");
choice12.addItem("TORPEDO ROOM");
choice12.addItem("AMR");
choice12.addItem("ERUL");
choice12.addItem("ERML");
choice12.addItem("TGLO BAY");
choice12.addItem("COND. BAY");
choice12.addItem("MSW BAY");
choice12.addItem("SHAFT ALLEY");
try {
    choice12.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice12);
choice12.setBounds(140,75,118,40);
label16 = new java.awt.Label("HOSE");
label16.setBounds(264,81,38,20);
panel2.add(label16);
choice13 = new java.awt.Choice();
choice13.addItem("FCML");
choice13.addItem("TORPEDO ROOM");
choice13.addItem("AMR");
choice13.addItem("ERUL FWD");
choice13.addItem("ERUL AFT");
choice13.addItem("TGLO BAY");
choice13.addItem("MSW BAY");
try {
    choice13.select(0);
}
catch (IllegalArgumentException e) { }

```

```

panel2.add(choice13);
choice13.setBounds(312,75,104,40);
textField3 = new java.awt.TextField();
textField3.setBounds(432,87,60,18);
panel2.add(textField3);
button3 = new java.awt.Button();
button3.setLabel("START");
button3.setBounds(516,87,50,18);
button3.setBackground(new Color(12632256));
panel2.add(button3);
choice14 = new java.awt.Choice();
choice14.addItem("CSES");
choice14.addItem("CONTROL");
choice14.addItem("NAV CENTER");
choice14.addItem("FCML");
choice14.addItem("TORPEDO ROOM");
choice14.addItem("AMR");
choice14.addItem("ERUL");
choice14.addItem("ERML");
choice14.addItem("TGLO BAY");
choice14.addItem("COND. BAY");
choice14.addItem("MSW BAY");
choice14.addItem("SHAFT ALLEY");
try {
    choice14.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice14);
choice14.setBounds(140,99,118,40);
label17 = new java.awt.Label("HOSE");
label17.setBounds(264,105,38,20);
panel2.add(label17);
choice15 = new java.awt.Choice();
choice15.addItem("FCML");
choice15.addItem("TORPEDO ROOM");
choice15.addItem("AMR");
choice15.addItem("ERUL FWD");
choice15.addItem("ERUL AFT");
choice15.addItem("TGLO BAY");
choice15.addItem("MSW BAY");
try {
    choice15.select(0);
}
catch (IllegalArgumentException e) { }
panel2.add(choice15);
choice15.setBounds(312,99,104,40);
textField4 = new java.awt.TextField();
textField4.setBounds(432,111,60,18);
panel2.add(textField4);
button4 = new java.awt.Button();
button4.setLabel("START");
button4.setBounds(516,111,50,18);
button4.setBackground(new Color(12632256));
panel2.add(button4);
label18 = new java.awt.Label("C");
label18.setBounds(123,79,12,24);

```

```

panel2.add(label18);
label19 = new java.awt.Label("D");
label19.setBounds(123,103,12,24);
panel2.add(label19);
textField6 = new java.awt.TextField();
textField6.setBounds(180,159,60,18);
panel2.add(textField6);
button6 = new java.awt.Button();
button6.setLabel("START");
button6.setBounds(264,159,50,18);
button6.setBackground(new Color(12632256));
panel2.add(button6);
textField7 = new java.awt.TextField();
textField7.setBounds(408,159,60,18);
panel2.add(textField7);
button7 = new java.awt.Button();
button7.setLabel("START");
button7.setBounds(492,159,50,18);
button7.setBackground(new Color(12632256));
panel2.add(button7);
label20 = new java.awt.Label("XO/MIC");
label20.setBounds(115,159,55,24);
panel2.add(label20);
label21 = new java.awt.Label("SMIC");
label21.setBounds(370,157,38,24);
panel2.add(label21);
label22 = new java.awt.Label("OBA");
label22.setBounds(192,140,40,20);
panel2.add(label22);
label24 = new java.awt.Label("OBA");
label24.setBounds(420,142,40,20);
panel2.add(label24);
label25 = new java.awt.Label("GROUND CHECKS");
label25.setBounds(24,195,125,12);
panel2.add(label25);
Group1 = new CheckboxGroup();
radioButton4 = new java.awt.Checkbox("SAT", Group1, false);
radioButton4.setBounds(180,195,60,10);
panel2.add(radioButton4);
radioButton5 = new java.awt.Checkbox("UNSAT", Group1,
false);
radioButton5.setBounds(264,195,65,10);
panel2.add(radioButton5);
Group2 = new CheckboxGroup();
radioButton6 = new java.awt.Checkbox("SAT", Group2, false);
radioButton6.setBounds(180,219,60,10);
panel2.add(radioButton6);
radioButton7 = new java.awt.Checkbox("UNSAT", Group2,
false);
radioButton7.setBounds(264,219,65,10);
panel2.add(radioButton7);
label26 = new java.awt.Label("ATMOSPHERE CHECKS");
label26.setBounds(24,219,155,12);
panel2.add(label26);
button15 = new java.awt.Button();
button15.setLabel("SUBMIT");

```

```

button15.setBounds(480,207,62,26);
button15.setBackground(new Color(12632256));
button15.addActionListener(this);
panel2.add(button15);
panel3 = new java.awt.Panel();
panel3.setLayout(null);
panel3.setVisible(false);
panel3.setBounds(12,33,576,268);
tabPanel1.add(panel3);
label1 = new java.awt.Label("REPORTING STATION");
label1.setBounds(10,5,145,23);
panel3.add(label1);
choice1 = new java.awt.Choice();
choice1.addItem("CSES");
choice1.addItem("CONTROL");
choice1.addItem("NAV CENTER");
choice1.addItem("FCML");
choice1.addItem("TORPEDO ROOM");
choice1.addItem("AMR");
choice1.addItem("ENGINE ROOM");
try {
    choice1.select(0);
}
catch (IllegalArgumentException e) { }
panel3.add(choice1);
choice1.setBounds(160,6,126,21);
label2 = new java.awt.Label("COMPARTMENT RIG :");
label2.setBounds(10,39,150,23);
panel3.add(label2);
button16 = new java.awt.Button();
button16.setLabel("SUBMIT");
button16.setBounds(480,207,62,26);
button16.setBackground(new Color(12632256));
button16.addActionListener(this);
panel3.add(button16);
checkbox7 = new java.awt.Checkbox("FIRE \\ GEN. EMERG");
checkbox7.setBounds(150,73,144,15);
panel3.add(checkbox7);
checkbox8 = new java.awt.Checkbox("SNORKEL \\ VENT.");
checkbox8.setBounds(150,107,132,15);
panel3.add(checkbox8);
checkbox9 = new java.awt.Checkbox("RFRE");
checkbox9.setBounds(150,141,60,15);
panel3.add(checkbox9);
panel4 = new java.awt.Panel();
panel4.setLayout(null);
panel4.setVisible(false);
panel4.setBounds(12,33,576,268);
tabPanel1.add(panel4);
label23 = new java.awt.Label("MAN IN CHARGE");
label23.setBounds(24,20,132,24);
panel4.add(label23);
choice16 = new java.awt.Choice();
choice16.addItem("LCDR DIORIO");
choice16.addItem("LCDR WOLD");
choice16.addItem("LT MUGGLEWORTH");

```

```

choice16.addItem("LT JONES");
choice16.addItem("LT DWYER");
choice16.addItem("LTjg INDELOCATO");
choice16.addItem("ETC SMITH");
choice16.addItem("ETC FOSTER");
choice16.addItem("MMC ALEMAN");

try {
    choice16.select(0);
}
catch (IllegalArgumentException e) { }
panel4.add(choice16);
choice16.setBounds(168,15,145,40);
label27 = new java.awt.Label("OBA TIMES :");
label27.setBounds(24,51,95,20);
panel4.add(label27);
label28 = new java.awt.Label("HOSE TEAM A");
label28.setBounds(108,75,99,20);
panel4.add(label28);
label29 = new java.awt.Label("HOSE TEAM B");
label29.setBounds(108,99,99,20);
panel4.add(label29);
label30 = new java.awt.Label("HOSE TEAM C");
label30.setBounds(108,123,99,20);
panel4.add(label30);
label31 = new java.awt.Label("HOSE TEAM D");
label31.setBounds(108,147,99,20);
panel4.add(label31);
label40 = new java.awt.Label("OBA TIME");
label40.setBounds(228,51,64,20);
panel4.add(label40);
textField5 = new java.awt.TextField();
textField5.setBounds(228,75,60,18);
panel4.add(textField5);
button5 = new java.awt.Button();
button5.setLabel("START");
button5.setBounds(312,75,50,18);
button5.setBackground(new Color(12632256));
panel4.add(button5);
textField8 = new java.awt.TextField();
textField8.setBounds(228,99,60,18);
panel4.add(textField8);
button8 = new java.awt.Button();
button8.setLabel("START");
button8.setBounds(312,99,50,18);
button8.setBackground(new Color(12632256));
panel4.add(button8);
textField9 = new java.awt.TextField();
textField9.setBounds(228,123,60,18);
panel4.add(textField9);
button9 = new java.awt.Button();
button9.setLabel("START");
button9.setBounds(312,123,50,18);
button9.setBackground(new Color(12632256));
panel4.add(button9);
textField10 = new java.awt.TextField();

```

```

textField10.setBounds(228,147,60,18);
panel4.add(textField10);
button10 = new java.awt.Button();
button10.setLabel("START");
button10.setBounds(312,147,50,18);
button10.setBackground(new Color(12632256));
panel4.add(button10);
label41 = new java.awt.Label("XO / MIC");
label41.setBounds(108,195,84,20);
panel4.add(label41);
label42 = new java.awt.Label("SMIC");
label42.setBounds(108,219,84,20);
panel4.add(label42);
textField11 = new java.awt.TextField();
textField11.setBounds(228,195,60,18);
panel4.add(textField11);
button11 = new java.awt.Button();
button11.setLabel("START");
button11.setBounds(312,195,50,18);
button11.setBackground(new Color(12632256));
panel4.add(button11);
textField12 = new java.awt.TextField();
textField12.setBounds(228,219,60,18);
panel4.add(textField12);
button12 = new java.awt.Button();
button12.setLabel("START");
button12.setBounds(312,219,50,18);
button12.setBackground(new Color(12632256));
panel4.add(button12);
button17 = new java.awt.Button();
button17.setLabel("SUBMIT");
button17.setBounds(480,207,62,26);
button17.setBackground(new Color(12632256));
button17.addActionListener(this);
panel4.add(button17);
panel5 = new java.awt.Panel();
panel5.setLayout(null);
panel5.setBounds(12,33,576,268);
tabPanel1.add(panel5);
label43 = new java.awt.Label("ATMOSPHERE MONITORING");
label43.setBounds(24,15,185,24);
label43.setBackground(new Color(12632256));
panel5.add(label43);
choice17 = new java.awt.Choice();
choice17.addItem("FWD COMP");
choice17.addItem("ENGINEER COM");
try {
    choice17.select(0);
}
catch (IllegalArgumentException e) { }
panel5.add(choice17);
choice17.setBounds(216,15,135,40);
label44 = new java.awt.Label("O2 (130-220 TORR)");
label44.setBounds(24,87,120,12);
panel5.add(label44);
Group12 = new CheckboxGroup();

```

```

false);
    radioButton8 = new java.awt.Checkbox("SAT", Group12,
radioButton8.setBounds(180,87,60,10);
panel5.add(radioButton8);
radioButton9 = new java.awt.Checkbox("UNSAT", Group12,
false);
    radioButton9.setBounds(264,87,65,10);
panel5.add(radioButton9);
Group13 = new CheckboxGroup();
radioButton10 = new java.awt.Checkbox("SAT", Group13,
false);
    radioButton10.setBounds(180,111,60,10);
panel5.add(radioButton10);
radioButton11 = new java.awt.Checkbox("UNSAT", Group13,
false);
    radioButton11.setBounds(264,111,65,10);
panel5.add(radioButton11);
label45 = new java.awt.Label("H2 ( < 7.6 TORR)");
label45.setBounds(24,111,144,12);
panel5.add(label45);
label46 = new java.awt.Label("CO ( < 152 MTORR)");
label46.setBounds(24,159,125,12);
panel5.add(label46);
Group15 = new CheckboxGroup();
radioButton12 = new java.awt.Checkbox("SAT", Group15,
false);
    radioButton12.setBounds(180,159,60,10);
panel5.add(radioButton12);
radioButton13 = new java.awt.Checkbox("UNSAT", Group15,
false);
    radioButton13.setBounds(264,159,65,10);
panel5.add(radioButton13);
Group16 = new CheckboxGroup();
radioButton20 = new java.awt.Checkbox("SAT", Group16,
false);
    radioButton20.setBounds(180,183,60,10);
panel5.add(radioButton20);
radioButton21 = new java.awt.Checkbox("UNSAT", Group16,
false);
    radioButton21.setBounds(264,183,65,10);
panel5.add(radioButton21);
label47 = new java.awt.Label("R114/12(<1520 MTORR)");
label47.setBounds(24,183,153,12);
panel5.add(label47);
Group18 = new CheckboxGroup();
radioButton24 = new java.awt.Checkbox("SAT", Group18,
false);
    radioButton24.setBounds(180,207,60,10);
panel5.add(radioButton24);
radioButton25 = new java.awt.Checkbox("UNSAT", Group18,
false);
    radioButton25.setBounds(264,207,65,10);
panel5.add(radioButton25);
label49 = new java.awt.Label("OTTO FUEL(<0.1 PPM)");
label49.setBounds(24,207,145,12);
panel5.add(label49);

```

```

        label50 = new java.awt.Label("CO2 ( < 30 TORR)");
        label50.setBounds(24,135,125,12);
        panel5.add(label50);
        Group14 = new CheckboxGroup();
        radioButton26 = new java.awt.Checkbox("SAT", Group14,
false);

        radioButton26.setBounds(180,135,60,10);
        panel5.add(radioButton26);
        radioButton27 = new java.awt.Checkbox("UNSAT", Group14,
false);

        radioButton27.setBounds(264,135,65,10);
        panel5.add(radioButton27);
        button18 = new java.awt.Button();
        button18.setLabel("SUBMIT");
        button18.setBounds(480,200,62,26);
        button18.setBackground(new Color(12632256));
        button18.addActionListener(this);
        panel5.add(button18);
        button14 = new java.awt.Button();
        button14.setLabel("SUBMIT");
        button14.setBounds(372,200,62,26);
        button14.setBackground(new Color(12632256));
        add(button14);

    } //end init

```

```

//{{DECLARE_CONTROLS
symantec.itools.awt.TabPanel tabPanel1;
java.awt.Panel panel1;
java.awt.Label label3;
java.awt.Label label7;
java.awt.Label label8;
java.awt.Label label9;
java.awt.Label label10;
java.awt.Choice choice2;
java.awt.Choice choice5;
java.awt.Choice choice6;
java.awt.Choice choice7;
java.awt.Label label4;
java.awt.Label label5;
java.awt.Label label6;
java.awt.Choice choice3;
java.awt.Choice choice4;
java.awt.Button button13;
java.awt.Checkbox checkbox1;
java.awt.Checkbox checkbox2;
java.awt.Checkbox checkbox3;
java.awt.Checkbox checkbox4;
java.awt.Checkbox checkbox5;
java.awt.Checkbox checkbox6;
java.awt.Panel panel2;

```

```
java.awt.Label label11;  
java.awt.Choice choice8;  
java.awt.Label label12;  
java.awt.Choice choice9;  
java.awt.Label label13;  
java.awt.TextField textField1;  
java.awt.Button button1;  
java.awt.Label label14;  
java.awt.Choice choice10;  
java.awt.Label label15;  
java.awt.Choice choice11;  
java.awt.TextField textField2;  
java.awt.Button button2;  
java.awt.Choice choice12;  
java.awt.Label label16;  
java.awt.Choice choice13;  
java.awt.TextField textField3;  
java.awt.Button button3;  
java.awt.Choice choice14;  
java.awt.Label label17;  
java.awt.Choice choice15;  
java.awt.TextField textField4;  
java.awt.Button button4;  
java.awt.Label label18;  
java.awt.Label label19;  
java.awt.TextField textField6;  
java.awt.Button button6;  
java.awt.TextField textField7;  
java.awt.Button button7;  
java.awt.Label label20;  
java.awt.Label label21;  
java.awt.Label label22;  
java.awt.Label label24;  
java.awt.Label label25;  
java.awt.Checkbox radioButton4;  
CheckboxGroup Group1;  
java.awt.Checkbox radioButton5;  
java.awt.Checkbox radioButton6;  
CheckboxGroup Group2;  
java.awt.Checkbox radioButton7;  
java.awt.Label label26;  
java.awt.Button button15;  
java.awt.Panel panel3;  
java.awt.Label label1;  
java.awt.Choice choice1;  
java.awt.Label label2;  
java.awt.Button button16;  
java.awt.Checkbox checkbox7;  
java.awt.Checkbox checkbox8;  
java.awt.Checkbox checkbox9;  
java.awt.Panel panel4;  
java.awt.Label label23;  
java.awt.Choice choice16;  
java.awt.Label label27;  
java.awt.Label label28;  
java.awt.Label label29;
```

```
java.awt.Label label30;
java.awt.Label label31;
java.awt.Label label32;
java.awt.Label label33;
java.awt.Label label34;
java.awt.Label label35;
java.awt.Label label36;
java.awt.Label label37;
java.awt.Label label38;
java.awt.Label label39;
java.awt.Label label40;
java.awt.TextField textField5;
java.awt.Button button5;
java.awt.TextField textField8;
java.awt.Button button8;
java.awt.TextField textField9;
java.awt.Button button9;
java.awt.TextField textField10;
java.awt.Button button10;
java.awt.Label label41;
java.awt.Label label42;
java.awt.TextField textField11;
java.awt.Button button11;
java.awt.TextField textField12;
java.awt.Button button12;
java.awt.Button button17;
java.awt.Panel panel5;
java.awt.Label label43;
java.awt.Choice choice17;
java.awt.Label label44;
java.awt.Checkbox radioButton8;
CheckboxGroup Group12;
java.awt.Checkbox radioButton9;
java.awt.Checkbox radioButton10;
CheckboxGroup Group13;
java.awt.Checkbox radioButton11;
java.awt.Label label45;
java.awt.Label label46;
java.awt.Checkbox radioButton12;
CheckboxGroup Group15;
java.awt.Checkbox radioButton13;
java.awt.Checkbox radioButton20;
CheckboxGroup Group16;
java.awt.Checkbox radioButton21;
java.awt.Label label47;
java.awt.Label label48;
java.awt.Checkbox radioButton22;
CheckboxGroup Group17;
java.awt.Checkbox radioButton23;
java.awt.Checkbox radioButton24;
CheckboxGroup Group18;
java.awt.Checkbox radioButton25;
java.awt.Label label49;
java.awt.Label label50;
java.awt.Checkbox radioButton26;
CheckboxGroup Group14;
```

```

        java.awt.Checkbox radioButton27;
        java.awt.Button button18;
        java.awt.Button button14;
        java.awt.Label label300;
        java.awt.Choice choice200;

        Thread      watchThread, watchThread1, watchThread2,
watchThread3, watchThread4;
        Thread      watchThread5, watchThread6, watchThread7,
watchThread8;
        Thread      watchThread9, watchThread10, watchThread11;
        Stopwatch   stopWatch, stopWatch1, stopWatch2, stopWatch3,
stopWatch4, stopWatch5;
        Stopwatch   stopWatch6, stopWatch7, stopWatch8, stopWatch9,
stopWatch10, stopWatch11;
        Date startDate, startDate1, startDate2, startDate3, startDate4,
startDate5;
        Date startDate6, startDate7, startDate8, startDate9, startDate10,
startDate11;

```

```

    public void run()
    {
        Thread currentthread = Thread.currentThread();

        while(true){// currentthread == outputThread) {
            try {
                currentthread.sleep(500);
            }
            catch (InterruptedException e) {}

            if ( Thread.currentThread().getName().equals("thread1")){
                stopWatch = new Stopwatch(textField1,startDate);
                stopWatch.nit();
            } //end if

            if ( Thread.currentThread().getName().equals("thread2")){
                stopWatch1 = new Stopwatch(textField2,startDate1);
            } // end if

            if ( Thread.currentThread().getName().equals("thread3")){
                stopWatch2 = new Stopwatch(textField3,startDate2);
            } // end if

            if ( Thread.currentThread().getName().equals("thread4")){
                stopWatch3 = new Stopwatch(textField4,startDate3);
            } //end if

            if ( Thread.currentThread().getName().equals("thread5")){
                stopWatch4 = new Stopwatch(textField6,startDate4);
            } // end if

            if ( Thread.currentThread().getName().equals("thread6")){
                stopWatch5 = new Stopwatch(textField7,startDate5);
            } // end if

```

```

        if ( Thread.currentThread().getName().equals("thread7")){
            stopWatch6 = new StopWatch(textField5,startDate6);
        } // end if

        if ( Thread.currentThread().getName().equals("thread8")){
            stopWatch7 = new StopWatch(textField8,startDate7);
        } // end if

        if ( Thread.currentThread().getName().equals("thread9")){
            stopWatch8 = new StopWatch(textField9,startDate8);
        } // end if

        if ( Thread.currentThread().getName().equals("thread10")){
            stopWatch9 = new StopWatch(textField10,startDate9);
        } // end if

        if ( Thread.currentThread().getName().equals("thread11")){
            stopWatch10 = new StopWatch(textField11,startDate10);
        } // end if

        if ( Thread.currentThread().getName().equals("thread12")){
            stopWatch11 = new StopWatch(textField12,startDate11);
        } // end if

    } //end while
} //end method run

```

```

// DATABASE VARIABLES
TextField theStatus      = new TextField(64);
JDBC01 theJDBC = new JDBC01(theStatus);

```

```

public void actionPerformed (ActionEvent event)
{
    Object source = event.getSource();

    if (source == button13) {

        boolean temp = false;

        try {
            theJDBC.executeQuery("SELECT CN FROM Tab1");
            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

            while (tokens.hasMoreTokens()){
                if (tokens.nextToken().equals("1")){
                    temp = true;
                } //end if
            } //END WHILE
        } //END TRY
        catch (SQLException sql) { ; }

        try {
            if (temp) {
                theJDBC.executeUpdate("UPDATE Tab1 "

```

```

+ "SET FIRE_IN =" + "'" + choice2.getSelectedItemAt()
+ "'"
+ ",FIRE_CLASS =" + "'" +
choice200.getSelectedItemAt() + "'"
+ ",FIRE_IS =" + "'" + choice5.getSelectedItemAt() +
"'"
+ ",MAN_CHAR =" + "'" + choice3.getSelectedItemAt() +
"'"
+ ",PERS =" + String.valueOf(checkbox1.getState())
+ ",OBA =" + String.valueOf(checkbox2.getState())
+ ",DC_EQ =" +
String.valueOf(checkbox4.getState())//3
+ ",NTFI =" +
String.valueOf(checkbox5.getState())//4
+ ",FFE =" +
String.valueOf(checkbox3.getState())//5
+ ",DAM_EQU =" + "'" + choice4.getSelectedItemAt() +
"'"
+ ",WATCH =" + "'" + choice6.getSelectedItemAt() +
"'"
+ ",HOSE =" + "'" + choice7.getSelectedItemAt() + "'"
+ ",SAT =" + String.valueOf(checkbox6.getState())
+ " WHERE CN = 1 " );
    } //end IF
} //END TRY
catch (SQLException sqlex) { }

} //end if

if (source == button15) {

    boolean temp = false;

    try {

        theJDBC.executeQuery("SELECT CN FROM Tab2");
        StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

        while (tokens.hasMoreTokens()){
            if (tokens.nextToken().equals("1")){
                temp = true;
            } //end if
        } //END WHILE
    } //END TRY
    catch (SQLException sql) { ; }

    try {
        String radiocheck1 = "N/A";
        String radiocheck2 = "N/A ";
        String t1 = textField1.getText();
        String t2 = textField2.getText();
        String t3 = textField3.getText();
        String t4 = textField4.getText();
        String t6 = textField6.getText();

```

```

String t7 = textField7.getText();

if ( radioButton4.getState()== true)
    radiocheck1 = "SAT";
else if ( radioButton5.getState()== true)
    radiocheck1 = "UNSAT";
if ( radioButton6.getState()== true)
    radiocheck2 = "SAT";
else if ( radioButton7.getState()== true)
    radiocheck2 = "UNSAT";

if ( t1.equals(""))
    t1 = "N/A";
if ( t2.equals(""))
    t2 = "N/A";
if ( t3.equals(""))
    t3 = "N/A";
if ( t4.equals(""))
    t4 = "N/A";
if ( t6.equals(""))
    t6 = "N/A";
if ( t7.equals(""))
    t7 = "N/A";

if (temp) {

    theJDBC.executeUpdate("UPDATE Tab2 "
+ "SET TEAM_A =" + "'" + choice8.getSelectedItemAt
+ ",HOSE_A =" + "'" + choice9.getSelectedItemAt
+ ",TEAM_B =" + "'" + choice10.getSelectedItemAt
+ ",HOSE_B =" + "'" + choice11.getSelectedItemAt
+ ",TEAM_C =" + "'" + choice12.getSelectedItemAt
+ ",HOSE_C =" + "'" + choice13.getSelectedItemAt
+ ",TEAM_D =" + "'" + choice14.getSelectedItemAt
+ ",HOSE_D =" + "'" + choice15.getSelectedItemAt

+ ",OBA_TIMEA =" + "'" + t1 + "'"
+ ",OBA_TIMEB =" + "'" + t2 + "'"
+ ",OBA_TIMEC =" + "'" + t3 + "'"
+ ",OBA_TIMED =" + "'" + t4 + "'"
+ ",OBA_TIMEXO =" + "'" + t6 + "'"
+ ",OBA_TIMESMIC =" + "'" + t7 + "'"
+ ",ATMCHK =" + "'" + radiocheck1 + "'"
+ ",GRNDCHK =" + "'" + radiocheck2 + "'"
+ " WHERE CN = 1 " );
} //end IF
} //END TRY

```

```

        catch (SQLException sqlex) { }
    } //end if

    if (source == button16) {

        boolean temp = false;

        try {

            theJDBC.executeQuery("SELECT CN FROM Tab3");
            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

            while (tokens.hasMoreTokens()){
                if (tokens.nextToken().equals("1")){
                    temp = true;
                } //end if
            } //END WHILE
        } //END TRY
        catch (SQLException sql) { ; }

        try {
            if (temp) {
                theJDBC.executeUpdate("UPDATE Tab3 "
+ "SET REP_STA = " + "'" + choice1.getSelectedItem()
+ "'"
+ ", FIRE_EMERG = " +
String.valueOf(checkbox7.getState())
+ ", SNORKEL = " +
String.valueOf(checkbox8.getState())
+ ", RFRE = " + String.valueOf(checkbox9.getState())
+ " WHERE CN = 1 " );
            } //end IF
        } //END TRY
        catch (SQLException sqlex) { }
    } //end if

    if (source == button17) {

        boolean temp = false;

        try {

            theJDBC.executeQuery("SELECT CN FROM Tab4");
            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

            while (tokens.hasMoreTokens()){
                if (tokens.nextToken().equals("1")){
                    temp = true;
                } //end if
            } //END WHILE
        } //END TRY
        catch (SQLException sql) { ; }
    }

```

```

try {

    String t5 = textField5.getText();
    String t8 = textField8.getText();
    String t9 = textField9.getText();
    String t10 = textField10.getText();
    String t11 = textField11.getText();
    String t12 = textField12.getText();

    if ( t5.equals(""))
        t5 = "N/A";
    if ( t8.equals(""))
        t8 = "N/A";
    if ( t9.equals(""))
        t9 = "N/A";
    if ( t10.equals(""))
        t10 = "N/A";
    if ( t11.equals(""))
        t11 = "N/A";
    if ( t12.equals(""))
        t12 = "N/A";

    if (temp) {

        theJDBC.executeUpdate("UPDATE Tab4 "
+ "SET MAN_IN =" + "'" + choice16.getSelectedItem()
+ "'"
+ ",HOSET_A =" + "'" + t5 + "'"
+ ",HOSET_B =" + "'" + t8 + "'"
+ ",HOSET_C =" + "'" + t9 + "'"
+ ",HOSET_D =" + "'" + t10 + "'"
+ ",XO_MIC =" + "'" + t11 + "'"
+ ",SMIC =" + "'" + t12 + "'"
+ " WHERE CN = 1 " );
    } //end IF
} //END TRY
catch (SQLException sqlex) { }
} //end if

if (source == button18) {

    boolean temp = false;

    try {

        theJDBC.executeQuery("SELECT CN FROM Tab5");
        StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

        while (tokens.hasMoreTokens()){
            if (tokens.nextToken().equals("1")){
                temp = true;
            } //end if
        } //END WHILE
    } //END TRY
}

```

```

        catch (SQLException sql) { ; }

    try {
        String radiocheck1 = "N/A";
        String radiocheck2 = "N/A ";
        String radiocheck3 = "N/A ";
        String radiocheck4 = "N/A ";
        String radiocheck5 = "N/A ";
        String radiocheck6 = "N/A ";

        if ( radioButton8.getState()== true)
            radiocheck1 = "SAT";
        else if ( radioButton9.getState()== true)
            radiocheck1 = "UNSAT";

        if ( radioButton10.getState()== true)
            radiocheck2 = "SAT";
        else if ( radioButton11.getState()== true)
            radiocheck2 = "UNSAT";

        if ( radioButton12.getState()== true)
            radiocheck3 = "SAT";
        else if ( radioButton13.getState()== true)
            radiocheck3 = "UNSAT";

        if ( radioButton20.getState()== true)
            radiocheck4 = "SAT";
        else if ( radioButton21.getState()== true)
            radiocheck4 = "UNSAT";

        if ( radioButton24.getState()== true)
            radiocheck5 = "SAT";
        else if ( radioButton25.getState()== true)
            radiocheck5 = "UNSAT";

        if ( radioButton26.getState()== true)
            radiocheck6 = "SAT";
        else if ( radioButton27.getState()== true)
            radiocheck6 = "UNSAT";

        if (temp) {
            theJDBC.executeUpdate("UPDATE Tab5 "
                + "SET ATM_MON =" + "'" +
choice17.getSelectedItemAt() + "'"
                + ",O2 =" + "'" + radiocheck1 + "'"
                + ",H2 =" + "'" + radiocheck2 + "'"
                + ",CO2 =" + "'" + radiocheck6 + "'"
                + ",CO =" + "'" + radiocheck3 + "'"
                + ",R114 =" + "'" + radiocheck4 + "'"
                + ",OTTO_FUEL =" + "'" + radiocheck5 + "'"
                + " WHERE CN = 1 " );
        } //end IF
    } //END TRY
    catch (SQLException sqlex) { }
} //end if

```

```

    } // end actionPerformed class

    // handle button events (CHANGES THE STATE OF THE "START" BUTTONS)

    public boolean action (Event e, Object o) {

        if ( e.target instanceof Button) {
            if ( (e. target == button1) &&
(button1.getLabel()=="START") ){
                button1.setLabel("RESET");
                startDate = new Date();
                watchThread = new Thread(this, "thread1");
                watchThread.setDaemon(false);
                watchThread.setPriority(4);
                watchThread.start();
            } //end if
            else if( (e. target == button1) &&
(button1.getLabel()=="RESET") ){
                button1.setLabel("START");
                watchThread.stop();
                textField1.setText("");
            } //end else if

            if ( e. target == button2 &&
(button2.getLabel()=="START")){
                button2.setLabel("RESET");
                startDate1 = new Date();
                watchThread1 = new Thread(this, "thread2");
                watchThread1.setDaemon(false);
                watchThread1.setPriority(4);
                watchThread1.start();
            } //end if
            else if( (e. target == button2) &&
(button2.getLabel()=="RESET") ){
                button2.setLabel("START");
                watchThread1.stop();
                textField2.setText("");
            } //end elseif

            if ( e. target == button3 &&
(button3.getLabel()=="START")) {
                button3.setLabel("RESET");
                startDate2 = new Date();
                watchThread2 = new Thread(this, "thread3");
                watchThread2.setPriority(4);
                watchThread2.start();
            } //end if
            else if( (e. target == button3) &&
(button3.getLabel()=="RESET") ){
                button3.setLabel("START");
                watchThread2.stop();
                textField3.setText("");
            } //end else if

```

```

        if ( e. target == button4 &&
(button4.getLabel()=="START")) {
            button4.setLabel("RESET");
            startDate3 = new Date();
            watchThread3 = new Thread(this, "thread4");
            watchThread3.setPriority(4);
            watchThread3.start();
        } //end if
        else if( (e. target == button4) &&
(button4.getLabel()=="RESET") ){
            button4.setLabel("START");
            watchThread3.stop();
            textField4.setText("");
        } //end else if

        if ( e. target == button5 &&
(button5.getLabel()=="START")) {
            button5.setLabel("RESET");
            startDate6 = new Date();
            watchThread6 = new Thread(this, "thread7");
            watchThread6.setPriority(4);
            watchThread6.start();
        } //end if
        else if( (e. target == button5) &&
(button5.getLabel()=="RESET") ) {
            button5.setLabel("START");
            watchThread6.stop();
            textField5.setText("");
        } //end else if

        if ( e. target == button6 &&
(button6.getLabel()=="START")){
            button6.setLabel("RESET");
            startDate4 = new Date();
            watchThread4 = new Thread(this, "thread5");
            watchThread4.setPriority(4);
            watchThread4.start();
        } //end if
        else if( (e. target == button6) &&
(button6.getLabel()=="RESET") ) {
            button6.setLabel("START");
            watchThread4.stop();
            textField6.setText("");
        } // end else

        if ( e. target == button7 &&
(button7.getLabel()=="START")) {
            button7.setLabel("RESET");
            startDate5 = new Date();
            watchThread5 = new Thread(this, "thread6");
            watchThread5.setPriority(4);
            watchThread5.start();
        } //end if
        else if( (e. target == button7) &&
(button7.getLabel()=="RESET") ) {
            button7.setLabel("START");

```

```

        watchThread5.stop();
        textField7.setText("");
    } // end else

    if ( e. target == button8 && (button8.getLabel()=="START"))
    {
        button8.setLabel("RESET");
        startDate7 = new Date();
        watchThread7 = new Thread(this, "thread8");
        watchThread7.setPriority(4);
        watchThread7.start();
    } //end if
    else if( (e. target == button8) &&
(button8.getLabel()=="RESET") ) {
        button8.setLabel("START");
        watchThread7.stop();
        textField8.setText("");
    } //end else if

    if ( e. target == button9 && (button9.getLabel()=="START"))
    {
        button9.setLabel("RESET");
        startDate8 = new Date();
        watchThread8 = new Thread(this, "thread9");
        watchThread8.setPriority(4);
        watchThread8.start();
    } //end if
    else if( (e. target == button9) &&
(button9.getLabel()=="RESET") ) {
        button9.setLabel("START");
        watchThread8.stop();
        textField9.setText("");
    } //end else if

    if ( e. target == button10 &&
(button10.getLabel()=="START")) {
        button10.setLabel("RESET");
        startDate9 = new Date();
        watchThread9 = new Thread(this, "thread10");
        watchThread9.setPriority(4);
        watchThread9.start();
    } //end if
    else if( (e. target == button10) &&
(button10.getLabel()=="RESET") ) {
        button10.setLabel("START");
        watchThread9.stop();
        textField10.setText("");
    } //end else if

    if ( e. target == button11 &&
(button11.getLabel()=="START")){
        button11.setLabel("RESET");
        startDate10 = new Date();
        watchThread10 = new Thread(this, "thread11");
        watchThread10.setPriority(4);
        watchThread10.start();

```

```

        } //end if
        else if( (e. target == button11) &&
(button11.getLabel()=="RESET") ) {
            button11.setLabel("START");
            watchThread10.stop();
            textField11.setText("");
        } //end else if

        if ( e. target == button12 &&
(button12.getLabel()=="START")) {
            button12.setLabel("RESET");
            startDate11 = new Date();
            watchThread11 = new Thread(this, "thread12");
            watchThread11.setPriority(4);
            watchThread11.start();
        } //end if
        else if( (e. target == button12) &&
(button12.getLabel()=="RESET") ) {
            button12.setLabel("START");
            watchThread11.stop();
            textField12.setText("");
        } //end else if

        if (e.id == Event.WINDOW_DESTROY ) {

            watchThread.stop();

        } //end if
    } //end if
    return true;
} //end action method

public void handleException(Exception e)
{
    e.printStackTrace( );
}

} //end class

```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. UTILITY CLASSES

```
// Stopwatch
// Used in new Attack Submarine Damage Control Applets and Servlets
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Provides a timer for OBA times.
//
// Contains no classified or sensitive data
//
// Date : 03 Oct. 1999
```

```
import java.awt.*;
import java.util.Date;
```

```
public class Stopwatch
{
```

```
    long startTime;
    Long difference;
    Date startDate;
    Date now;
    int hr;
    int mn;
    int sc;
    Integer hour;
    Integer minute;
    Integer second;
    String message;
    boolean watchActivated = true;
    TextField myfield = new TextField();
```

```
    public Stopwatch(TextField tf, Date d){
        myfield = tf;
        startDate = d;
        nit();
    }
```

```
    public void stop()
    {
        watchActivated = false;
    }
```

```
    public void nit()
    {

        startTime = startDate.getTime();

        if (watchActivated) {

            now = new Date();
```

```

        difference = new Long(now.getTime() - startTime);
        sc = difference.intValue() / 1000;

        mn = sc / 60;
        hr = mn / 60;
        sc = sc % 60;
        mn = mn % 60;
        second = new Integer(sc);
        minute = new Integer(mn);
        hour = new Integer(hr);
        message = minute.toString() + ":" + second.toString();
        myfield.setText(message);
    }
}

}

*****

// JDB01 CClass
// CS3773 Java as a second Language Final Project
// Version 1.0
// Kurt Rothenhaus
// CLASS automates a number of usefull functions that the sorter
// uses to display its results
// Contains no classified or sensitive data
// Programmer wishes to thank Java; how to program by
// Dietel & Dietel and
// SQL programming in Java by Bill McCarthy for their valuable
// code which I unabashadly used.
// Applet can be viewed at:
// http://web.nps.navy.mil/~kjrothen

import java.awt.*;
import java.awt.event.*;
import java.util.Properties;
import java.net.URL;
import j102.sql.*; //Creates abstract bridge for socket/JDBC
import java.applet.*;

public class JDB01
{
    Connection        theConnection = null; // the JDBC bridge
    DatabaseMetaData  theDBMetaData = null;
    Statement          theStatement   = null;
    ResultSet          theResultSet   = null;
    ResultSetMetaData theMetaData     = null;
    TextField theStatus;

    public JDB01(TextField status)

```

```

    {
        theStatus = status;
    }

    // This procedure opens the "socket" to the ODBC bridge
    public void openConnection()throws SQLException
    {
        try
        {
            IDSDriver drv = new j102.sql.IDSDriver(); // New ODBC
driver instantiated
            String url = "jdbc:ids://131.120.27.67:12/conn?dsn='db1'";
            Connection theConnection = drv.connect(url,null); //connect
is really a java.sql
            // }
            // catch (Exception e){ }

            //Download the database attributes and create a result set.
            theDBMetaData
            = theConnection.getMetaData( );
            theStatement
            = theConnection.createStatement( );
            theResultSet = null;
            theMetaData = null;
            theStatus.setText("Status: OK");
        }
        catch (SQLException sql)
        {
            handleError(sql);
        }
    }

    // Closes the connection to the database when the program is quit.
    public void closeConnection( )throws SQLException
    {
        try
        {
            if (theConnection != null)
                theConnection.close( );
        }
        catch (SQLException sql) { handleError(sql); }
    }

    // Executes the desired query from a string passed in.
    public void executeQuery(String sql)throws SQLException
    {
        if (theResultSet != null)
            theResultSet.close( );
    }

```

```

        theResultSet = theStatement.executeQuery(sql);
        if (theResultSet != null)
            theMetaData = theResultSet.getMetaData( );
    }

```

//Allows the user to update the database (used in viewscreen)

```

public int executeUpdate(String sql)throws SQLException
{
    if (theResultSet != null)
        theResultSet.close();
    theResultSet = null;
    theMetaData = null;
    int result = theStatement.executeUpdate(sql);
    return result;
}

```

// Takes the results of the query and turns it into a long string.

```

public String dumpResult( )throws SQLException
{
    String result = "";
    try
    {
        int column_count = theMetaData.getColumnCount( );
        while (theResultSet.next( ))
        {
            boolean first = true;
            for (int i = 1; i <= column_count; i++)
            {
                if (!first) result += ", ";
                result += theResultSet.getString(i);
                first = false;
            }
            result += "\n";
        }
    }
    catch (SQLException sql) { handleError(sql); }
    return result;
}

```

// inserts the fields in order

```

String getFieldList(String [ ] fields)
{
    String result = "(";
    boolean first = true;
    for (int i = 0; i < fields.length; i++)
    {
        if (!first) result += ", ";
        first = false;
        result += fields[i];
    }
}

```

```

    }
    result += ") ";
    return result;
}

//
String getValueList(String [ ] values, boolean [ ] isQuoted)
{
    String result = "VALUES (";
    boolean first = true;
    for (int i = 0; i < values.length; i++)
    {
        if (!first) result += ", ";
        first = false;
        String value = values[i];
        if (isQuoted[i])
        {
            result += "'";

            // double any embedded single quotes
            int j;
            while ((j = value.indexOf('\')) >= 0)
            {
                if (j > 0)
                {
                    result += value.substring(0, j);
                }
                result += "'";
                if (value.length( ) > j + 1)
                {
                    value = value.substring(j + 1);
                }
                else
                {
                    value = "";
                }
            }
            result += value + "'";
        }
        else
        {
            result += value;
        }
    }
    result += ") ";
    return result;
}

String getNonNullString(int col)
throws SQLException
{
    return nonNull(theResultSet.getString(col));
}

```

```

String nonNull(String s)
{
    if (s != null) return s;
    return "";
}

// Handles errors that arise from SQL misrep.

public void handleError(Throwable t) throws SQLException
{
    theStatus.setText("Error: " + t.getMessage( ));
    t.printStackTrace( );
    throw new SQLException(t.getMessage( ));
}

} // end class

```

APPENDIX C. DAMAGE CONTROL CLIENT SERVLET CODE

```
// DC Client Servlet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/dc_ServCover.htm
// Date : 10 Oct. 1999

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class DC_ClientServlet extends HttpServlet
{

    public void doPost(HttpServletRequest req,
                       HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());

        // Print the HTML header
        out.println("<html>");
        out.println("<head>");
        out.println("<title>DC Report</title>");
        out.println("</head>");
        out.println("<h2><center>");
        out.println("DC Updated: Press Back button to return");
        out.println("</center></h2>");
        out.println("<br>");

        String values[];

        // Get the location
        String FIRE_IN = "";
        values = req.getParameterValues("Selection");
        if (values != null) {
            FIRE_IN = values[0];
        }
        out.println("FIRE_IN=" + FIRE_IN + "<br>");
    }
}
```

```

// Get the location
String FIRE_CLASS = "";
values = req.getParameterValues("Selection14");
if (values != null) {
    FIRE_CLASS = values[0];
}
out.println("FIRE_CLASS=" + FIRE_CLASS + "<br>");

// get state of fire
String FIRE_IS = "";
values = req.getParameterValues("Selection9");
if (values != null) {
    FIRE_IS = values[0];
}
out.println("FIRE_IS=" + FIRE_IS + "<br>");

//Get man in charge
String MAN_CHAR = "";
values = req.getParameterValues("Selection2");
if (values != null) {
    MAN_CHAR = values[0];
}
out.println("MAN_CHAR=" + MAN_CHAR + "<br>");
// Get the assist
String PERS = "no";
String PERS1 = "0";
values = req.getParameterValues("CheckBox");
if (values != null) {
    PERS = "yes";
    PERS1 = "1";
}
out.println("PERS=" + PERS + "<br>");
// Get the assist
String OBA = "no";
int OBA1 = 0;
values = req.getParameterValues("CheckBox4");
if (values != null) {
    OBA = "yes";
    OBA1 = 1;
}
out.println("OBA=" + OBA + "<br>");
// Get the assist
String DC_EQ = "no";
int DC_EQ1 = 0;
values = req.getParameterValues("CheckBox6");
if (values != null) {
    DC_EQ = "yes";
    DC_EQ1 = 1;
}
out.println("DC_EQ=" + DC_EQ + "<br>");
// Get the assist
String NTFI = "no";
int NTFI1 = 0;
values = req.getParameterValues("CheckBox7");
if (values != null) {
    NTFI = "yes";

```

```

        NTFI1 = 1;
    }
    out.println("NTFI=" + NTFI + "<br>");
    // Get the assist
    String FFE = "no";
    int FFE1 = 0;
    values = req.getParameterValues("CheckBox5");
    if (values != null) {
        FFE = "yes";
        FFE1 = 1;
    }
    out.println("FFE=" + FFE + "<br>");
    String SAT = "no";
    int SAT1 = 0;
    values = req.getParameterValues("CheckBox12");
    if (values != null) {
        SAT = "yes";
        SAT1 = 1;
    }
    out.println("SAT=" + SAT + "<br>");
    SAT = String.valueOf(SAT);
    //get damaged equipment
    String DAM_EQU = "";
    values = req.getParameterValues("Selection8");
    if (values != null) {
        DAM_EQU = values[0];
    }
    out.println("DAM_EQU=" + DAM_EQU + "<br>");
    //get watch
    String WATCH = "";
    values = req.getParameterValues("Selection10");
    if (values != null) {
        WATCH = values[0];
    }
    out.println("WATCH=" + WATCH + "<br>");
    //get hose status
    String HOSE = "";
    values = req.getParameterValues("Selection11");
    if (values != null) {
        HOSE = values[0];
    }
    out.println("HOSE=" + HOSE + "<br>");

    try {

        int casualty = 1;
        Connection con = null;
        Statement stmt = null;

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
        con =
        DriverManager.getConnection("jdbc:odbc:db1",null,null);
        stmt = con.createStatement();
        stmt.executeUpdate("UPDATE Tab1 "
        + "SET FIRE_IN =" + "" + FIRE_IN + ""


```

```

        + ",FIRE_CLASS =" + "'" + FIRE_CLASS + "'"
        + ",FIRE_IS =" + "'" + FIRE_IS + "'"
        + ",MAN_CHAR =" + "'" + MAN_CHAR + "'"
        + ",PERS =" + "'" + PERS1 + "'"
        + ",OBA =" + "'" + OBA1 + "'"
        + ",DC_EQ =" + "'" + DC_EQ1 + "'"
        + ",NTFI =" + "'" + NTFI1 + "'"
        + ",FFE =" + "'" + FFE1 + "'"
        + ",SAT =" + "'" + SAT1 + "'"
        + ",DAM_EQU =" + "'" + DAM_EQU + "'"
        + ",WATCH =" + "'" + WATCH + "'"
        + ",HOSE =" + "'" + HOSE + "'"
        + " WHERE CN=1");
    }
    catch (Exception ex) {
        out.println("Exception!");
        ex.printStackTrace(out);
    }
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
}

}

*****

// Central Servlet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/dc_ServCover.html
// Date : 12 Oct. 1999

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class dc_centServ extends HttpServlet
{
    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response

```

```

java.io.PrintWriter out =
    new java.io.PrintWriter(resp.getOutputStream());

// Print the HTML header
out.println("<html>");
out.println("<head>");
out.println("<title>DC CENTRAL </title>");
out.println("</head>");
out.println("<h2><center>");
out.println("DC Updated: Press Back button to return");
out.println("</center></h2>");
out.println("<br>");

String values[];

// Get location of team A
String TEAM_A = "";
values = req.getParameterValues("Selection");
if (values != null) {
    TEAM_A = values[0];
}
out.println("TEAM_A=" + TEAM_A + "<br>");

// get location of hose A
String HOSE_A = "";
values = req.getParameterValues("Selection2");
if (values != null) {
    HOSE_A = values[0];
}
out.println("HOSE_A=" + HOSE_A + "<br>");

// Get location of team B
String TEAM_B = "";
values = req.getParameterValues("Selection6");
if (values != null) {
    TEAM_B = values[0];
}
out.println("TEAM_B=" + TEAM_B + "<br>");

// get location of hose B
String HOSE_B = "";
values = req.getParameterValues("Selection7");
if (values != null) {
    HOSE_B = values[0];
}
out.println("HOSE_B=" + HOSE_B + "<br>");

// Get location of team C
String TEAM_C = "";
values = req.getParameterValues("Selection11");
if (values != null) {
    TEAM_C = values[0];
}
out.println("TEAM_C=" + TEAM_C + "<br>");

// get location of hose C

```

```

String HOSE_C = "";
values = req.getParameterValues("Selection12");
if (values != null) {
    HOSE_C = values[0];
}
out.println("HOSE_C=" + HOSE_C + "<br>");

// Get location of team D
String TEAM_D = "";
values = req.getParameterValues("Selection16");
if (values != null) {
    TEAM_D = values[0];
}
out.println("TEAM_D=" + TEAM_D + "<br>");

// get location of hose D
String HOSE_D = "";
values = req.getParameterValues("Selection17");
if (values != null) {
    HOSE_D = values[0];
}
out.println("HOSE_D=" + HOSE_D + "<br>");

// Get the state of atmosphere
String ATMCHK = "N/A";

values = req.getParameterValues("Radio31");
if (values != null) {
    ATMCHK = values[0];
}
if (ATMCHK.equals("Radio310"))
    ATMCHK = "SAT";
if (ATMCHK.equals("Radio311"))
    ATMCHK = "UNSAT";

out.println("ATMCHK=" + ATMCHK + "<br>");

// Get the state of ground
String GRNDCHK = "N/A";

values = req.getParameterValues("Radio32");
if (values != null) {
    GRNDCHK = values[0];
}
if (GRNDCHK.equals("Radio320"))
    GRNDCHK = "SAT";
if (GRNDCHK.equals("Radio321"))
    GRNDCHK = "UNSAT";

out.println("GRNDCHK=" + GRNDCHK + "<br>");
//*****
/* // Get the assist
String PERS = "no";
String PERS1 = "0";
values = req.getParameterValues("CheckBox");

```

```

    if (values != null) {
        PERS = values[0];
        PERS1 = "1";
    }
    out.println("PERS=" + PERS + "<br>");
    // Get the assist
    String OBA = "no";
    int OBA1 = 0;
    values = req.getParameterValues("CheckBox4");
    if (values != null) {
        OBA = values[0];
        OBA1 = 1;
    }
    out.println("OBA=" + OBA + "<br>");
    // Get the assist
    String DC_EQ = "no";
    int DC_EQ1 = 0;
    values = req.getParameterValues("CheckBox6");
    if (values != null) {
        DC_EQ = values[0];
        DC_EQ1 = 1;
    }
    out.println("DC_EQ=" + DC_EQ + "<br>");
    // Get the assist
    String NTFI = "no";
    int NTFI1 = 0;
    values = req.getParameterValues("CheckBox7");
    if (values != null) {
        NTFI = values[0];
        NTFI1 = 1;
    }
    out.println("NTFI=" + NTFI + "<br>");
    // Get the assist
    String FFE = "no";
    int FFE1 = 0;
    values = req.getParameterValues("CheckBox5");
    if (values != null) {
        FFE = values[0];
        FFE1 = 1;
    }
    out.println("FFE=" + FFE + "<br>");
    String SAT = "no";
    int SAT1 = 0;
    values = req.getParameterValues("CheckBox12");
    if (values != null) {
        SAT = values[0];
        SAT1 = 1;
    }
    out.println("SAT=" + SAT + "<br>");
    SAT = String.valueOf(SAT);
    //get damaged equipment
    String DAM_EQU = "";
    values = req.getParameterValues("Selection8");
    if (values != null) {
        DAM_EQU = values[0];
    }

```

```

out.println("DAM_EQU=" + DAM_EQU + "<br>");
//get watch
String WATCH = "";
values = req.getParameterValues("Selection10");
if (values != null) {
    WATCH = values[0];
}
out.println("WATCH=" + WATCH + "<br>");
//get hose status
String HOSE = "";
values = req.getParameterValues("Selection11");
if (values != null) {
    HOSE = values[0];
}
out.println("HOSE=" + HOSE + "<br>");*/
String OBA_TIMEA = "";
String OBA_TIMEB = "";
String OBA_TIMEC = "";
String OBA_TIMED = "";
String OBA_TIME XO = "";
String OBA_TIMESMIC = "";

try {

    int casualty = 1;
    Connection con = null;
    Statement stmt = null;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
    con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
    stmt = con.createStatement();
    stmt.executeUpdate("UPDATE Tab2 "
+ "SET TEAM_A =" + "'" + TEAM_A + "'"
+ ",HOSE_A =" + "'" + HOSE_A + "'"
+ ",TEAM_B =" + "'" + TEAM_B + "'"
+ ",HOSE_B =" + "'" + HOSE_B + "'"
+ ",TEAM_C =" + "'" + TEAM_C + "'"
+ ",HOSE_C =" + "'" + HOSE_C + "'"
+ ",TEAM_D =" + "'" + TEAM_D + "'"
+ ",HOSE_D =" + "'" + HOSE_D + "'"
+ ",OBA_TIMEA =" + "'" + OBA_TIMEA + "'" //+
dummy inputs
+ ",OBA_TIMEB =" + "'" + OBA_TIMEB + "'"
+ ",OBA_TIMEC =" + "'" + OBA_TIMEC + "'"
+ ",OBA_TIMED =" + "'" + OBA_TIMED + "'"
+ ",OBA_TIME XO =" + "'" + OBA_TIME XO + "'"
+ ",OBA_TIMESMIC =" + "'" + OBA_TIMESMIC + "'"
+ ",ATMCHK =" + "'" + ATMCHK + "'"
+ ",GRNDCHK =" + "'" + GRNDCHK + "'"
+ " WHERE CN=1");
}
catch (Exception ex) {
    out.println("Exception!");
    ex.printStackTrace(out);
}

```

```

    }
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
}

}
// Watch Stander Servlet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/dc_ServCover.html
// Date : 30 Sept. 1999

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class dc_watchServ extends HttpServlet
{
    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        boolean t1 = false;
        boolean t2 = false;
        boolean t3 = false;

        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());

        // Print the HTML header
        out.println("<html>");
        out.println("<head>");
        out.println("<title>DC WATCHSTANDER</title>");
        out.println("</head>");
        out.println("<h2><center>");
        out.println("DC Updated: Press Back button to return");
        out.println("</center></h2>");
        out.println("<br>");

        String values[];

        // Get the station ID

```

```

String REP_STA = "";

values = req.getParameterValues("Selection");
if (values != null) {
    REP_STA = values[0];
}
out.println("REP_STA=" + REP_STA + "<br>");

// Get the assist
String FIRE_EMERG = "no";
int f1 = 0;
values = req.getParameterValues("CheckBox");
if (values != null) {
    FIRE_EMERG = "yes";
    f1 = 1;
}

out.println("FIRE_EMERG=" + FIRE_EMERG + "<br>");

// Get the assist
String SNORKEL = "no";
int f2 = 0;

values = req.getParameterValues("CheckBox3");
if (values != null) {
    SNORKEL = "yes";
    f2 = 1;
}

out.println(" SNORKEL=" + SNORKEL + "<br>");

// Get the assist
String RFRE = "no";
int f3 = 0;

values = req.getParameterValues("CheckBox4");
if (values != null) {
    RFRE = "yes";
    f3 = 1;
}

out.println("RFRE=" + RFRE + "<br>");

//chage the states of checkbox to boolean
if ( FIRE_EMERG.equals("CheckBox"))
    t1 = true;
if ( SNORKEL.equals("CheckBox"))
    t2 = true;
if ( RFRE.equals("CheckBox"))
    t3 = true;

try {

    int casualty = 1;
    Connection con = null;
    Statement stmt = null;

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
    con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
    stmt = con.createStatement();
    stmt.executeUpdate("UPDATE Tab3 "
+ "SET REP_STA =" + "'" + REP_STA + "'"
+ ",FIRE_EMERG =" + f1
+ ",SNORKEL =" + f2
+ ",RFRE =" + f3
+ " WHERE CN=1");
    }
    catch (Exception ex) {
        out.println("Exception!");
        ex.printStackTrace(out);
    }
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
}

}

*****

// Staging Servlet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Applet can be viewed at:
// http://131.120.27.67:8080//dc_ServCover.htm
// Date : 15 Oct. 1999

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class dc_stageServ extends HttpServlet
{

    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =

```

```

        new java.io.PrintWriter(resp.getOutputStream());

// Print the HTML header
out.println("<html>");
out.println("<head>");
out.println("<title>DC STAGE </title>");
out.println("</head>");
out.println("<h2><center>");
out.println("DC Updated: Press Back button to return");
out.println("</center></h2>");
out.println("<br>");

String values[];

// Get the location
String MAN_IN = "";
values = req.getParameterValues("Selection");
if (values != null) {
    MAN_IN = values[0];
}
out.println("MAN_IN=" + MAN_IN + "<br>");

/* // get state of fire
String HOSE_A = "";
values = req.getParameterValues("Selection2");
if (values != null) {
    HOSE_A = values[0];
}
out.println("HOSE_A=" + HOSE_A + "<br>");
//Get man in charge
String TEAM_B = "";
values = req.getParameterValues("Selection6");
if (values != null) {
    TEAM_B = values[0];
}
out.println("TEAM_B=" + TEAM_B + "<br>");

// get state of fire
String HOSE_B = "";
values = req.getParameterValues("Selection7");
if (values != null) {
    HOSE_B = values[0];
}
out.println("HOSE_B=" + HOSE_B + "<br>");
//Get man in charge
String TEAM_C = "";
values = req.getParameterValues("Selection11");
if (values != null) {
    TEAM_C = values[0];
}
out.println("TEAM_C=" + TEAM_C + "<br>");

// get state of fire
String HOSE_C = "";
values = req.getParameterValues("Selection12");
if (values != null) {

```

```

        HOSE_C = values[0];
    }
    out.println("HOSE_C=" + HOSE_C + "<br>");
    //Get man in charge
    String TEAM_D = "";
    values = req.getParameterValues("Selection16");
    if (values != null) {
        TEAM_D = values[0];
    }
    out.println("TEAM_D=" + TEAM_D + "<br>");

    // get state of fire
    String HOSE_D = "";
    values = req.getParameterValues("Selection17");
    if (values != null) {
        HOSE_D = values[0];
    }
    out.println("HOSE_D=" + HOSE_D + "<br>");
    //*****
    // Get the assist
    String PERS = "no";
    String PERS1 = "0";
    values = req.getParameterValues("CheckBox");
    if (values != null) {
        PERS = values[0];
        PERS1 = "1";
    }
    out.println("PERS=" + PERS + "<br>");
    // Get the assist
    String OBA = "no";
    int OBA1 = 0;
    values = req.getParameterValues("CheckBox4");
    if (values != null) {
        OBA = values[0];
        OBA1 = 1;
    }
    out.println("OBA=" + OBA + "<br>");
    // Get the assist
    String DC_EQ = "no";
    int DC_EQ1 = 0;
    values = req.getParameterValues("CheckBox6");
    if (values != null) {
        DC_EQ = values[0];
        DC_EQ1 = 1;
    }
    out.println("DC_EQ=" + DC_EQ + "<br>");
    // Get the assist
    String NTFI = "no";
    int NTFI1 = 0;
    values = req.getParameterValues("CheckBox7");
    if (values != null) {
        NTFI = values[0];
        NTFI1 = 1;
    }
    out.println("NTFI=" + NTFI + "<br>");
    // Get the assist

```

```

String FFE = "no";
int FFE1 = 0;
values = req.getParameterValues("CheckBox5");
if (values != null) {
    FFE = values[0];
    FFE1 = 1;
}
out.println("FFE=" + FFE + "<br>");
String SAT = "no";
int SAT1 = 0;
values = req.getParameterValues("CheckBox12");
if (values != null) {
    SAT = values[0];
    SAT1 = 1;
}
out.println("SAT=" + SAT + "<br>");
SAT = String.valueOf(SAT);
//get damaged equipment
String DAM_EQU = "";
values = req.getParameterValues("Selection8");
if (values != null) {
    DAM_EQU = values[0];
}
out.println("DAM_EQU=" + DAM_EQU + "<br>");
//get watch
String WATCH = "";
values = req.getParameterValues("Selection10");
if (values != null) {
    WATCH = values[0];
}
out.println("WATCH=" + WATCH + "<br>");
//get hose status
String HOSE = "";
values = req.getParameterValues("Selection11");
if (values != null) {
    HOSE = values[0];
}
out.println("HOSE=" + HOSE + "<br>");*/
String HOSET_A = "";
String HOSET_B = "";
String HOSET_C = "";
String HOSET_D = "";
String XO_MIC = "";
String SMIC = "";

```

```

try {

```

```

    int casualty = 1;
    Connection con = null;
    Statement stmt = null;

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

```

```

        con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
        stmt = con.createStatement();
        stmt.executeUpdate("UPDATE Tab4 "
+ "SET MAN_IN =" + "'" + MAN_IN + "'"
+ ",HOSET_A =" + "'" + HOSET_A + "'" // dummy inputs
+ ",HOSET_B =" + "'" + HOSET_B + "'"
+ ",HOSET_C =" + "'" + HOSET_C + "'"
+ ",HOSET_D =" + "'" + HOSET_D + "'"
+ ",XO_MIC =" + "'" + XO_MIC + "'"
+ ",SMIC =" + "'" + SMIC + "'"
+ " WHERE CN=1");
    }
    catch (Exception ex) {
        out.println("Exception!");
        ex.printStackTrace(out);
    }
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
}
}

```

```

// DC Client Servlet
// New Attack Submarine Damage Control Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view DC actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servletlet can be viewed at:
// http://131.120.27.67:8080/dc_ServCover.htm
// Date : 30 Sept. 1999

```

```

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

```

```

public class dc_atmServ extends HttpServlet
{

```

```

    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());
    }
}

```

```

// Print the HTML header
out.println("<html>");
out.println("<head>");
out.println("<title>DC ATM. MONITORING </title>");
out.println("</head>");
out.println("<h2><center>");
out.println("DC Updated: Press Back button to return");
out.println("</center></h2>");
out.println("<br>");

String values[];

// Get the location
String ATM_MON = "";

values = req.getParameterValues("Selection");
if (values != null) {
    ATM_MON = values[0];
}
out.println("ATM_MON=" + ATM_MON + "<br>");

// Get the feedback of variable modified in the database
String O2 = "N/A";

values = req.getParameterValues("Radio3");
if (values != null) {
    O2 = values[0];
}
if (O2.equals("Radio30"))
    O2 = "SAT";
if (O2.equals("Radio31"))
    O2 = "UNSAT";
out.println("O2=" + O2 + "<br>");

// Get the feedback of variable modified in the database
String H2 = "N/A";

values = req.getParameterValues("Radio4");
if (values != null) {
    H2 = values[0];
}
if (H2.equals("Radio40"))
    H2 = "SAT";
if (H2.equals("Radio41"))
    H2 = "UNSAT";
out.println("H2=" + H2 + "<br>");

// Get the feedback of variable modified in the database
String CO2 = "N/A";

values = req.getParameterValues("Radio5");
if (values != null) {
    CO2 = values[0];
}
if (CO2.equals("Radio50"))
    CO2 = "SAT";

```

```

    if ( CO2.equals("Radio51"))
        CO2 = "UNSAT";
    out.println("CO2=" + CO2 + "<br>");

    // Get the feedback of variable modified in the database
    String CO = "N/A";

    values = req.getParameterValues("Radio6");
    if (values != null) {
        CO = values[0];
    }
    if ( CO.equals("Radio60"))
        CO = "SAT";
    if ( CO.equals("Radio61"))
        CO = "UNSAT";
    out.println("CO=" + CO + "<br>");

    // Get the feedback of variable modified in the database
    String R114 = "N/A";

    values = req.getParameterValues("Radio7");
    if (values != null) {
        R114 = values[0];
    }
    if ( R114.equals("Radio70"))
        R114 = "SAT";
    if ( R114.equals("Radio71"))
        R114 = "UNSAT";
    out.println("R114=" + R114 + "<br>");

    // Get the feedback of variable modified in the database
    String OTTO_FUEL = "N/A";

    values = req.getParameterValues("Radio8");
    if (values != null) {
        OTTO_FUEL = values[0];
    }
    if ( OTTO_FUEL.equals("Radio80"))
        OTTO_FUEL = "SAT";
    if ( OTTO_FUEL.equals("Radio81"))
        OTTO_FUEL = "UNSAT";
    out.println("OTTO_FUEL=" + OTTO_FUEL + "<br>");

    // Database connection and modification
    try {

        int casualty = 1;
        Connection con = null;
        Statement stmt = null;

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
        con =
        DriverManager.getConnection("jdbc:odbc:db1",null,null);
        stmt = con.createStatement();
        stmt.executeUpdate("UPDATE Tab5 "
            + "SET ATM_MON = " + "'" + ATM_MON + "'"

```

```

        + ",O2 =" + "'" + O2 + "'"
        + ",H2 =" + "'" + H2 + "'"
        + ",CO2 =" + "'" + CO2 + "'"
        + ",CO =" + "'" + CO + "'"
        + ",R114 =" + "'" + R114 + "'"
        + ",OTTO_FUEL =" + "'" + OTTO_FUEL + "'"
        + " WHERE CN=1");
    }
    catch (Exception ex) {
        out.println("Exception!");
        ex.printStackTrace(out);
    }
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
}
}

```

APPENDIX D. LOG TAKING CLIENT APPLET CODE

```
// LOG TAKING Client Applet
// New Attack Submarine
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Applet allows to modify a SQL complaint database through a database
// server(IDS server) by using JDBC connectivity.
// Applet was written for USN vehicles for specified US Naval Vessels
// (Simulated)
// Allows watchstanders sending information by using mobile computer
// devices over wireless LANs.
// Contains no classified or sensitive data
// Applet can be viewed at:
// http://131.120.27.67:12/log.html
// Date : 27 Oct. 1999
```

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import j102.sql.*;
import java.util.Date;
import symantec.itools.awt.TabPanel;
import symantec.itools.awt.BorderPanel;
import symantec.itools.awt.ScrollingPanel;
```

```
public class Applet1 extends Applet implements ActionListener
{

    public void start()
    {
        super.start();

        try
        {
            theJDBC.openConnection();
        } // end if
        catch (Exception e)
        {
            handleException(e);
        } // end catch
    } // end method start

    public void stop()
    {
        try
```

```

    {
        theJDBC.closeConnection();

    } // end try
    catch (Exception e)
    {
        handleException(e);
    } // end catch
} // end method stop


public void init()
{
    // Take out this line if you don't use
    symantec.itools.net.RelativeURL or
    symantec.itools.awt.util.StatusScroller
    //symantec.itools.lang.Context.setApplet(this);

    // This code is automatically generated by Visual Cafe when
    you add
    // components to the visual environment. It instantiates
    and initializes
    // the components. To modify the code, only use code syntax
    that matches
    // what Visual Cafe can generate, or Visual Cafe may be
    unable to back
    // parse your Java file into its visual environment.
    //{{INIT_CONTROLS
    setLayout(null);
    setSize(426,266);
    setFont(new Font("Dialog", Font.PLAIN, 12));
    setForeground(new Color(0));
    setBackground(new Color(12632256));
    tabPanel1 = new symantec.itools.awt.TabPanel();
    try {
        java.lang.String[] tempString = new
java.lang.String[3];
        tempString[0] = new java.lang.String("AEF Logs");
        tempString[1] = new java.lang.String("Atmosphere
Control Logs");
        tempString[2] = new java.lang.String("Remarks");
        tabPanel1.setPanelLabels(tempString);
    }
    catch(java.beans.PropertyVetoException e) { }
    try {
        tabPanel1.setCurrentPanelNdx(0);
    }
    catch(java.beans.PropertyVetoException e) { }
    tabPanel1.setBounds(72,36,660,480);
    tabPanel1.setBackground(new Color(12632256));
    add(tabPanel1);
    panel1 = new java.awt.Panel();
    panel1.setLayout(null);
    panel1.setVisible(false);
    panel1.setBounds(12,33,636,436);

```

```

panel1.setBackground(new Color(12632256));
tabPanel1.add(panel1);
tabPanel2 = new symantec.itools.awt.TabPanel();
try {
    java.lang.String[] tempString = new
java.lang.String[2];
    tempString[0] = new java.lang.String("400 Cycle
Sets");
    tempString[1] = new java.lang.String("Detectors &
Indicators");
    tabPanel2.setPanelLabels(tempString);
}
catch(java.beans.PropertyVetoException e) { }
try {
    tabPanel2.setCurrentPanelNdx(1);
}
catch(java.beans.PropertyVetoException e) { }
tabPanel2.setBounds(12,27,611,380);
tabPanel2.setFont(new Font("Dialog", Font.PLAIN, 12));
tabPanel2.setBackground(new Color(12632256));
panel1.add(tabPanel2);
panel3 = new java.awt.Panel();
panel3.setLayout(null);
panel3.setVisible(false);
panel3.setBounds(12,33,587,336);
tabPanel2.add(panel3);
button1 = new java.awt.Button();
button1.addActionListener(this);
button1.setLabel("Submit");
button1.setBounds(504,303,68,19);
button1.setBackground(new Color(12632256));
panel3.add(button1);
button2 = new java.awt.Button();
button2.addActionListener(this);
button2.setLabel("Retrieve");
button2.setBounds(420,303,69,19);
button2.setBackground(new Color(12632256));
panel3.add(button2);
textField1 = new java.awt.TextField();
textField1.setEditable(false);
textField1.setBounds(10,63,67,20);
textField1.setFont(new Font("Dialog", Font.PLAIN, 10));
textField1.setForeground(new Color(0));
textField1.setBackground(new Color(16777215));
panel3.add(textField1);
textField2 = new java.awt.TextField();
textField2.setBounds(10,87,67,20);
textField2.setFont(new Font("Dialog", Font.PLAIN, 10));
textField2.setBackground(new Color(16777215));
panel3.add(textField2);
textField3 = new java.awt.TextField();
textField3.setBounds(10,111,67,20);
textField3.setFont(new Font("Dialog", Font.PLAIN, 10));
textField3.setBackground(new Color(16777215));
panel3.add(textField3);
textField4 = new java.awt.TextField();

```

```

textField4.setBounds(10,135,67,20);
textField4.setFont(new Font("Dialog", Font.PLAIN, 10));
textField4.setBackground(new Color(16777215));
panel3.add(textField4);
textField5 = new java.awt.TextField();
textField5.setBounds(10,159,67,20);
textField5.setFont(new Font("Dialog", Font.PLAIN, 10));
textField5.setBackground(new Color(16777215));
panel3.add(textField5);
textField6 = new java.awt.TextField();
textField6.setBounds(10,183,67,20);
textField6.setFont(new Font("Dialog", Font.PLAIN, 10));
textField6.setBackground(new Color(16777215));
panel3.add(textField6);
textField7 = new java.awt.TextField();
textField7.setBounds(96,63,45,20);
textField7.setFont(new Font("Dialog", Font.PLAIN, 10));
textField7.setBackground(new Color(-128));
panel3.add(textField7);
textField8 = new java.awt.TextField();
textField8.setBounds(144,63,45,20);
textField8.setFont(new Font("Dialog", Font.PLAIN, 10));
textField8.setBackground(new Color(-8323200));
panel3.add(textField8);
textField9 = new java.awt.TextField();
textField9.setBounds(192,63,45,20);
textField9.setFont(new Font("Dialog", Font.PLAIN, 10));
textField9.setBackground(new Color(-128));
panel3.add(textField9);
textField10 = new java.awt.TextField();
textField10.setBounds(240,63,45,20);
textField10.setFont(new Font("Dialog", Font.PLAIN, 10));
textField10.setBackground(new Color(-8323200));
panel3.add(textField10);
textField11 = new java.awt.TextField();
textField11.setBounds(288,63,45,20);
textField11.setFont(new Font("Dialog", Font.PLAIN, 10));
textField11.setBackground(new Color(-128));
panel3.add(textField11);
textField12 = new java.awt.TextField();
textField12.setBounds(348,63,45,20);
textField12.setFont(new Font("Dialog", Font.PLAIN, 10));
textField12.setBackground(new Color(-8323200));
panel3.add(textField12);
textField13 = new java.awt.TextField();
textField13.setBounds(396,63,45,20);
textField13.setFont(new Font("Dialog", Font.PLAIN, 10));
textField13.setBackground(new Color(-128));
panel3.add(textField13);
textField14 = new java.awt.TextField();
textField14.setBounds(444,63,45,20);
textField14.setFont(new Font("Dialog", Font.PLAIN, 10));
textField14.setBackground(new Color(-8323200));
panel3.add(textField14);
textField15 = new java.awt.TextField();
textField15.setBounds(492,63,45,20);

```

```

textField15.setFont(new Font("Dialog", Font.PLAIN, 10));
textField15.setBackground(new Color(-128));
panel3.add(textField15);
textField16 = new java.awt.TextField();
textField16.setBounds(540,63,45,20);
textField16.setFont(new Font("Dialog", Font.PLAIN, 10));
textField16.setBackground(new Color(-8323200));
panel3.add(textField16);
textField17 = new java.awt.TextField();
textField17.setBounds(96,87,45,20);
textField17.setFont(new Font("Dialog", Font.PLAIN, 10));
textField17.setBackground(new Color(-128));
panel3.add(textField17);
textField18 = new java.awt.TextField();
textField18.setBounds(144,87,45,20);
textField18.setFont(new Font("Dialog", Font.PLAIN, 10));
textField18.setBackground(new Color(-8323200));
panel3.add(textField18);
textField19 = new java.awt.TextField();
textField19.setBounds(192,87,45,20);
textField19.setFont(new Font("Dialog", Font.PLAIN, 10));
textField19.setBackground(new Color(-128));
panel3.add(textField19);
textField20 = new java.awt.TextField();
textField20.setBounds(240,87,45,20);
textField20.setFont(new Font("Dialog", Font.PLAIN, 10));
textField20.setBackground(new Color(-8323200));
panel3.add(textField20);
textField21 = new java.awt.TextField();
textField21.setBounds(288,87,45,20);
textField21.setFont(new Font("Dialog", Font.PLAIN, 10));
textField21.setBackground(new Color(-128));
panel3.add(textField21);
textField22 = new java.awt.TextField();
textField22.setBounds(348,87,45,20);
textField22.setFont(new Font("Dialog", Font.PLAIN, 10));
textField22.setBackground(new Color(-8323200));
panel3.add(textField22);
textField23 = new java.awt.TextField();
textField23.setBounds(396,87,45,20);
textField23.setFont(new Font("Dialog", Font.PLAIN, 10));
textField23.setBackground(new Color(-128));
panel3.add(textField23);
textField24 = new java.awt.TextField();
textField24.setBounds(444,87,45,20);
textField24.setFont(new Font("Dialog", Font.PLAIN, 10));
textField24.setBackground(new Color(-8323200));
panel3.add(textField24);
textField25 = new java.awt.TextField();
textField25.setBounds(492,87,45,20);
textField25.setFont(new Font("Dialog", Font.PLAIN, 10));
textField25.setBackground(new Color(-128));
panel3.add(textField25);
textField26 = new java.awt.TextField();
textField26.setBounds(540,87,45,20);
textField26.setFont(new Font("Dialog", Font.PLAIN, 10));

```

```

textField26.setBackground(new Color(-8323200));
panel3.add(textField26);
textField27 = new java.awt.TextField();
textField27.setBounds(96,111,45,20);
textField27.setFont(new Font("Dialog", Font.PLAIN, 10));
textField27.setBackground(new Color(-128));
panel3.add(textField27);
textField28 = new java.awt.TextField();
textField28.setBounds(144,111,45,20);
textField28.setFont(new Font("Dialog", Font.PLAIN, 10));
textField28.setBackground(new Color(-8323200));
panel3.add(textField28);
textField29 = new java.awt.TextField();
textField29.setBounds(192,111,45,20);
textField29.setFont(new Font("Dialog", Font.PLAIN, 10));
textField29.setBackground(new Color(-128));
panel3.add(textField29);
textField30 = new java.awt.TextField();
textField30.setBounds(240,111,45,20);
textField30.setFont(new Font("Dialog", Font.PLAIN, 10));
textField30.setBackground(new Color(-8323200));
panel3.add(textField30);
textField31 = new java.awt.TextField();
textField31.setBounds(288,111,45,20);
textField31.setFont(new Font("Dialog", Font.PLAIN, 10));
textField31.setBackground(new Color(-128));
panel3.add(textField31);
textField32 = new java.awt.TextField();
textField32.setBounds(348,111,45,20);
textField32.setFont(new Font("Dialog", Font.PLAIN, 10));
textField32.setBackground(new Color(-8323200));
panel3.add(textField32);
textField33 = new java.awt.TextField();
textField33.setBounds(396,111,45,20);
textField33.setFont(new Font("Dialog", Font.PLAIN, 10));
textField33.setBackground(new Color(-128));
panel3.add(textField33);
textField34 = new java.awt.TextField();
textField34.setBounds(444,111,45,20);
textField34.setFont(new Font("Dialog", Font.PLAIN, 10));
textField34.setBackground(new Color(-8323200));
panel3.add(textField34);
textField35 = new java.awt.TextField();
textField35.setBounds(492,111,45,20);
textField35.setFont(new Font("Dialog", Font.PLAIN, 10));
textField35.setBackground(new Color(-128));
panel3.add(textField35);
textField36 = new java.awt.TextField();
textField36.setBounds(540,111,45,20);
textField36.setFont(new Font("Dialog", Font.PLAIN, 10));
textField36.setBackground(new Color(-8323200));
panel3.add(textField36);
textField37 = new java.awt.TextField();
textField37.setBounds(96,135,45,20);
textField37.setFont(new Font("Dialog", Font.PLAIN, 10));
textField37.setBackground(new Color(-128));

```

```

panel3.add(textField37);
textField38 = new java.awt.TextField();
textField38.setBounds(144,135,45,20);
textField38.setFont(new Font("Dialog", Font.PLAIN, 10));
textField38.setBackground(new Color(-8323200));
panel3.add(textField38);
textField39 = new java.awt.TextField();
textField39.setBounds(192,135,45,20);
textField39.setFont(new Font("Dialog", Font.PLAIN, 10));
textField39.setBackground(new Color(-128));
panel3.add(textField39);
textField40 = new java.awt.TextField();
textField40.setBounds(240,135,45,20);
textField40.setFont(new Font("Dialog", Font.PLAIN, 10));
textField40.setBackground(new Color(-8323200));
panel3.add(textField40);
textField41 = new java.awt.TextField();
textField41.setBounds(288,135,45,20);
textField41.setFont(new Font("Dialog", Font.PLAIN, 10));
textField41.setBackground(new Color(-128));
panel3.add(textField41);
textField42 = new java.awt.TextField();
textField42.setBounds(348,135,45,20);
textField42.setFont(new Font("Dialog", Font.PLAIN, 10));
textField42.setBackground(new Color(-8323200));
panel3.add(textField42);
textField43 = new java.awt.TextField();
textField43.setBounds(396,135,45,20);
textField43.setFont(new Font("Dialog", Font.PLAIN, 10));
textField43.setBackground(new Color(-128));
panel3.add(textField43);
textField44 = new java.awt.TextField();
textField44.setBounds(444,135,45,20);
textField44.setFont(new Font("Dialog", Font.PLAIN, 10));
textField44.setBackground(new Color(-8323200));
panel3.add(textField44);
textField45 = new java.awt.TextField();
textField45.setBounds(492,135,45,20);
textField45.setFont(new Font("Dialog", Font.PLAIN, 10));
textField45.setBackground(new Color(-128));
panel3.add(textField45);
textField46 = new java.awt.TextField();
textField46.setBounds(540,135,45,20);
textField46.setFont(new Font("Dialog", Font.PLAIN, 10));
textField46.setBackground(new Color(-8323200));
panel3.add(textField46);
textField47 = new java.awt.TextField();
textField47.setBounds(96,159,45,20);
textField47.setFont(new Font("Dialog", Font.PLAIN, 10));
textField47.setBackground(new Color(-128));
panel3.add(textField47);
textField48 = new java.awt.TextField();
textField48.setBounds(144,159,45,20);
textField48.setFont(new Font("Dialog", Font.PLAIN, 10));
textField48.setBackground(new Color(-8323200));
panel3.add(textField48);

```

```

textField49 = new java.awt.TextField();
textField49.setBounds(192,159,45,20);
textField49.setFont(new Font("Dialog", Font.PLAIN, 10));
textField49.setBackground(new Color(-128));
panel3.add(textField49);
textField50 = new java.awt.TextField();
textField50.setBounds(240,159,45,20);
textField50.setFont(new Font("Dialog", Font.PLAIN, 10));
textField50.setBackground(new Color(-8323200));
panel3.add(textField50);
textField51 = new java.awt.TextField();
textField51.setBounds(288,159,45,20);
textField51.setFont(new Font("Dialog", Font.PLAIN, 10));
textField51.setBackground(new Color(-128));
panel3.add(textField51);
textField52 = new java.awt.TextField();
textField52.setBounds(348,159,45,20);
textField52.setFont(new Font("Dialog", Font.PLAIN, 10));
textField52.setBackground(new Color(-8323200));
panel3.add(textField52);
textField53 = new java.awt.TextField();
textField53.setBounds(396,159,45,20);
textField53.setFont(new Font("Dialog", Font.PLAIN, 10));
textField53.setBackground(new Color(-128));
panel3.add(textField53);
textField54 = new java.awt.TextField();
textField54.setBounds(444,159,45,20);
textField54.setFont(new Font("Dialog", Font.PLAIN, 10));
textField54.setBackground(new Color(-8323200));
panel3.add(textField54);
textField55 = new java.awt.TextField();
textField55.setBounds(492,159,45,20);
textField55.setFont(new Font("Dialog", Font.PLAIN, 10));
textField55.setBackground(new Color(-128));
panel3.add(textField55);
textField56 = new java.awt.TextField();
textField56.setBounds(540,159,45,20);
textField56.setFont(new Font("Dialog", Font.PLAIN, 10));
textField56.setBackground(new Color(-8323200));
panel3.add(textField56);
textField57 = new java.awt.TextField();
textField57.setBounds(96,183,45,20);
textField57.setFont(new Font("Dialog", Font.PLAIN, 10));
textField57.setBackground(new Color(-128));
panel3.add(textField57);
textField58 = new java.awt.TextField();
textField58.setBounds(144,183,45,20);
textField58.setFont(new Font("Dialog", Font.PLAIN, 10));
textField58.setBackground(new Color(-8323200));
panel3.add(textField58);
textField59 = new java.awt.TextField();
textField59.setBounds(192,183,45,20);
textField59.setFont(new Font("Dialog", Font.PLAIN, 10));
textField59.setBackground(new Color(-128));
panel3.add(textField59);
textField60 = new java.awt.TextField();

```

```

textField60.setBounds(240,183,45,20);
textField60.setFont(new Font("Dialog", Font.PLAIN, 10));
textField60.setBackground(new Color(-8323200));
panel3.add(textField60);
textField61 = new java.awt.TextField();
textField61.setBounds(288,183,45,20);
textField61.setFont(new Font("Dialog", Font.PLAIN, 10));
textField61.setBackground(new Color(-128));
panel3.add(textField61);
textField62 = new java.awt.TextField();
textField62.setBounds(348,183,45,20);
textField62.setFont(new Font("Dialog", Font.PLAIN, 10));
textField62.setBackground(new Color(-8323200));
panel3.add(textField62);
textField63 = new java.awt.TextField();
textField63.setBounds(396,183,45,20);
textField63.setFont(new Font("Dialog", Font.PLAIN, 10));
textField63.setBackground(new Color(-128));
panel3.add(textField63);
textField64 = new java.awt.TextField();
textField64.setBounds(444,183,45,20);
textField64.setFont(new Font("Dialog", Font.PLAIN, 10));
textField64.setBackground(new Color(-8323200));
panel3.add(textField64);
textField65 = new java.awt.TextField();
textField65.setBounds(492,183,45,20);
textField65.setFont(new Font("Dialog", Font.PLAIN, 10));
textField65.setBackground(new Color(-128));
panel3.add(textField65);
textField66 = new java.awt.TextField();
textField66.setBounds(540,183,45,20);
textField66.setFont(new Font("Dialog", Font.PLAIN, 10));
textField66.setBackground(new Color(-8323200));
panel3.add(textField66);
label1 = new java.awt.Label("Time Period");
label1.setBounds(12,39,67,15);
label1.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label1);
label2 = new java.awt.Label("DC Amps");
label2.setBounds(96,39,45,15);
label2.setFont(new Font("Dialog", Font.PLAIN, 9));
panel3.add(label2);
label3 = new java.awt.Label("AC Amps");
label3.setBounds(144,39,45,15);
label3.setFont(new Font("Dialog", Font.PLAIN, 9));
panel3.add(label3);
label4 = new java.awt.Label("    Freq");
label4.setBounds(192,39,45,15);
label4.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label4);
label5 = new java.awt.Label("    Volts");
label5.setBounds(240,39,45,15);
label5.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label5);
label6 = new java.awt.Label("Grnd Det");
label6.setBounds(288,39,45,15);

```

```

label6.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label6);
label7 = new java.awt.Label("DC Amps");
label7.setBounds(348,39,45,15);
label7.setFont(new Font("Dialog", Font.PLAIN, 9));
panel3.add(label7);
label8 = new java.awt.Label("AC Amps");
label8.setBounds(396,39,45,15);
label8.setFont(new Font("Dialog", Font.PLAIN, 9));
panel3.add(label8);
label9 = new java.awt.Label("    Freq");
label9.setBounds(444,39,45,15);
label9.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label9);
label10 = new java.awt.Label("    Volts");
label10.setBounds(492,39,45,15);
label10.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label10);
label11 = new java.awt.Label("Grnd Det");
label11.setBounds(540,39,45,15);
label11.setFont(new Font("Dialog", Font.PLAIN, 10));
panel3.add(label11);
label12 = new java.awt.Label("                                NO.
1");
label12.setBounds(96,15,240,20);
label12.setFont(new Font("Dialog", Font.BOLD, 15));
panel3.add(label12);
label13 = new java.awt.Label("                                NO.
2");
label13.setBounds(348,15,240,20);
label13.setFont(new Font("Dialog", Font.BOLD, 15));
panel3.add(label13);
textField67 = new java.awt.TextField();
textField67.setBounds(96,243,45,20);
textField67.setFont(new Font("Dialog", Font.PLAIN, 10));
textField67.setBackground(new Color(16777215));
panel3.add(textField67);
textField68 = new java.awt.TextField();
textField68.setBounds(144,243,45,20);
textField68.setFont(new Font("Dialog", Font.PLAIN, 10));
textField68.setBackground(new Color(16777215));
panel3.add(textField68);
textField69 = new java.awt.TextField();
textField69.setBounds(192,243,45,20);
textField69.setFont(new Font("Dialog", Font.PLAIN, 10));
textField69.setBackground(new Color(16777215));
panel3.add(textField69);
textField70 = new java.awt.TextField();
textField70.setBounds(240,243,45,20);
textField70.setFont(new Font("Dialog", Font.PLAIN, 10));
textField70.setBackground(new Color(16777215));
panel3.add(textField70);
textField71 = new java.awt.TextField();
textField71.setBounds(288,243,45,20);
textField71.setFont(new Font("Dialog", Font.PLAIN, 10));
textField71.setBackground(new Color(16777215));

```

```

panel3.add(textField71);
textField72 = new java.awt.TextField();
textField72.setBounds(348,243,45,20);
textField72.setFont(new Font("Dialog", Font.PLAIN, 10));
textField72.setBackground(new Color(16777215));
panel3.add(textField72);
textField73 = new java.awt.TextField();
textField73.setBounds(396,243,45,20);
textField73.setFont(new Font("Dialog", Font.PLAIN, 10));
textField73.setBackground(new Color(16777215));
panel3.add(textField73);
textField74 = new java.awt.TextField();
textField74.setBounds(444,243,45,20);
textField74.setFont(new Font("Dialog", Font.PLAIN, 10));
textField74.setBackground(new Color(16777215));
panel3.add(textField74);
textField75 = new java.awt.TextField();
textField75.setBounds(492,243,45,20);
textField75.setFont(new Font("Dialog", Font.PLAIN, 10));
textField75.setBackground(new Color(16777215));
panel3.add(textField75);
textField76 = new java.awt.TextField();
textField76.setBounds(540,243,45,20);
textField76.setFont(new Font("Dialog", Font.PLAIN, 10));
textField76.setBackground(new Color(16777215));
panel3.add(textField76);
choice1 = new java.awt.Choice();
choice1.addItem("00:00");
choice1.addItem("01:00");
choice1.addItem("02:00");
choice1.addItem("03:00");
choice1.addItem("04:00");
choice1.addItem("05:00");
choice1.addItem("06:00");
choice1.addItem("07:00");
choice1.addItem("08:00");
choice1.addItem("09:00");
choice1.addItem("10:00");
choice1.addItem("11:00");
choice1.addItem("12:00");
choice1.addItem("13:00");
choice1.addItem("14:00");
choice1.addItem("15:00");
choice1.addItem("16:00");
choice1.addItem("17:00");
choice1.addItem("18:00");
choice1.addItem("19:00");
choice1.addItem("20:00");
choice1.addItem("21:00");
choice1.addItem("22:00");
choice1.addItem("23:00");
panel3.add(choice1);
choice1.setBounds(10,243,67,20);
choice1.setBackground(new Color(16777215));
panel4 = new java.awt.Panel();
panel4.setLayout(null);

```

```

panel4.setBounds(12,33,587,336);
panel4.setBackground(new Color(12632256));
tabPanel2.add(panel4);
textField77 = new java.awt.TextField();
textField77.setEditable(false);
textField77.setBounds(0,75,67,20);
textField77.setFont(new Font("Dialog", Font.PLAIN, 10));
textField77.setForeground(new Color(0));
textField77.setBackground(new Color(16777215));
panel4.add(textField77);
textField78 = new java.awt.TextField();
textField78.setBounds(0,99,67,20);
textField78.setFont(new Font("Dialog", Font.PLAIN, 10));
textField78.setBackground(new Color(16777215));
panel4.add(textField78);
textField79 = new java.awt.TextField();
textField79.setBounds(0,123,67,20);
textField79.setFont(new Font("Dialog", Font.PLAIN, 10));
textField79.setBackground(new Color(16777215));
panel4.add(textField79);
textField80 = new java.awt.TextField();
textField80.setBounds(0,147,67,20);
textField80.setFont(new Font("Dialog", Font.PLAIN, 10));
textField80.setBackground(new Color(16777215));
panel4.add(textField80);
textField81 = new java.awt.TextField();
textField81.setBounds(0,171,67,20);
textField81.setFont(new Font("Dialog", Font.PLAIN, 10));
textField81.setBackground(new Color(16777215));
panel4.add(textField81);
textField82 = new java.awt.TextField();
textField82.setBounds(0,195,67,20);
textField82.setFont(new Font("Dialog", Font.PLAIN, 10));
textField82.setBackground(new Color(16777215));
panel4.add(textField82);
textField83 = new java.awt.TextField();
textField83.setBounds(84,75,45,20);
textField83.setFont(new Font("Dialog", Font.PLAIN, 10));
textField83.setBackground(new Color(-128));
panel4.add(textField83);
textField84 = new java.awt.TextField();
textField84.setBounds(132,75,45,20);
textField84.setFont(new Font("Dialog", Font.PLAIN, 10));
textField84.setBackground(new Color(-8323200));
panel4.add(textField84);
textField85 = new java.awt.TextField();
textField85.setBounds(180,75,45,20);
textField85.setFont(new Font("Dialog", Font.PLAIN, 10));
textField85.setBackground(new Color(-128));
panel4.add(textField85);
textField86 = new java.awt.TextField();
textField86.setBounds(228,75,45,20);
textField86.setFont(new Font("Dialog", Font.PLAIN, 10));
textField86.setBackground(new Color(-8323200));
panel4.add(textField86);
textField87 = new java.awt.TextField();

```

```

textField87.setBounds(276,75,45,20);
textField87.setFont(new Font("Dialog", Font.PLAIN, 10));
textField87.setBackground(new Color(-128));
panel4.add(textField87);
textField88 = new java.awt.TextField();
textField88.setBounds(324,75,45,20);
textField88.setFont(new Font("Dialog", Font.PLAIN, 10));
textField88.setBackground(new Color(-8323200));
panel4.add(textField88);
textField89 = new java.awt.TextField();
textField89.setBounds(372,75,45,20);
textField89.setFont(new Font("Dialog", Font.PLAIN, 10));
textField89.setBackground(new Color(-128));
panel4.add(textField89);
textField90 = new java.awt.TextField();
textField90.setBounds(420,75,45,20);
textField90.setFont(new Font("Dialog", Font.PLAIN, 10));
textField90.setBackground(new Color(-8323200));
panel4.add(textField90);
textField91 = new java.awt.TextField();
textField91.setBounds(468,75,25,20);
textField91.setFont(new Font("Dialog", Font.PLAIN, 10));
textField91.setBackground(new Color(-128));
panel4.add(textField91);
textField92 = new java.awt.TextField();
textField92.setBounds(496,75,25,20);
textField92.setFont(new Font("Dialog", Font.PLAIN, 10));
textField92.setBackground(new Color(-8323200));
panel4.add(textField92);
textField93 = new java.awt.TextField();
textField93.setBounds(84,99,45,20);
textField93.setFont(new Font("Dialog", Font.PLAIN, 10));
textField93.setBackground(new Color(-128));
panel4.add(textField93);
textField94 = new java.awt.TextField();
textField94.setBounds(132,99,45,20);
textField94.setFont(new Font("Dialog", Font.PLAIN, 10));
textField94.setBackground(new Color(-8323200));
panel4.add(textField94);
textField95 = new java.awt.TextField();
textField95.setBounds(180,99,45,20);
textField95.setFont(new Font("Dialog", Font.PLAIN, 10));
textField95.setBackground(new Color(-128));
panel4.add(textField95);
textField96 = new java.awt.TextField();
textField96.setBounds(228,99,45,20);
textField96.setFont(new Font("Dialog", Font.PLAIN, 10));
textField96.setBackground(new Color(-8323200));
panel4.add(textField96);
textField97 = new java.awt.TextField();
textField97.setBounds(276,99,45,20);
textField97.setFont(new Font("Dialog", Font.PLAIN, 10));
textField97.setBackground(new Color(-128));
panel4.add(textField97);
textField98 = new java.awt.TextField();
textField98.setBounds(324,99,45,20);

```

```

textField98.setFont(new Font("Dialog", Font.PLAIN, 10));
textField98.setBackground(new Color(-8323200));
panel4.add(textField98);
textField99 = new java.awt.TextField();
textField99.setBounds(372,99,45,20);
textField99.setFont(new Font("Dialog", Font.PLAIN, 10));
textField99.setBackground(new Color(-128));
panel4.add(textField99);
textField100 = new java.awt.TextField();
textField100.setBounds(420,99,45,20);
textField100.setFont(new Font("Dialog", Font.PLAIN, 10));
textField100.setBackground(new Color(-8323200));
panel4.add(textField100);
textField101 = new java.awt.TextField();
textField101.setBounds(468,99,25,20);
textField101.setFont(new Font("Dialog", Font.PLAIN, 10));
textField101.setBackground(new Color(-128));
panel4.add(textField101);
textField102 = new java.awt.TextField();
textField102.setBounds(496,99,25,20);
textField102.setFont(new Font("Dialog", Font.PLAIN, 10));
textField102.setBackground(new Color(-8323200));
panel4.add(textField102);
textField103 = new java.awt.TextField();
textField103.setBounds(84,123,45,20);
textField103.setFont(new Font("Dialog", Font.PLAIN, 10));
textField103.setBackground(new Color(-128));
panel4.add(textField103);
textField104 = new java.awt.TextField();
textField104.setBounds(132,123,45,20);
textField104.setFont(new Font("Dialog", Font.PLAIN, 10));
textField104.setBackground(new Color(-8323200));
panel4.add(textField104);
textField105 = new java.awt.TextField();
textField105.setBounds(180,123,45,20);
textField105.setFont(new Font("Dialog", Font.PLAIN, 10));
textField105.setBackground(new Color(-128));
panel4.add(textField105);
textField106 = new java.awt.TextField();
textField106.setBounds(228,123,45,20);
textField106.setFont(new Font("Dialog", Font.PLAIN, 10));
textField106.setBackground(new Color(-8323200));
panel4.add(textField106);
textField107 = new java.awt.TextField();
textField107.setBounds(276,123,45,20);
textField107.setFont(new Font("Dialog", Font.PLAIN, 10));
textField107.setBackground(new Color(-128));
panel4.add(textField107);
textField108 = new java.awt.TextField();
textField108.setBounds(324,123,45,20);
textField108.setFont(new Font("Dialog", Font.PLAIN, 10));
textField108.setBackground(new Color(-8323200));
panel4.add(textField108);
textField109 = new java.awt.TextField();
textField109.setBounds(372,123,45,20);
textField109.setFont(new Font("Dialog", Font.PLAIN, 10));

```

```

textField109.setBackground(new Color(-128));
panel4.add(textField109);
textField110 = new java.awt.TextField();
textField110.setBounds(420,123,45,20);
textField110.setFont(new Font("Dialog", Font.PLAIN, 10));
textField110.setBackground(new Color(-8323200));
panel4.add(textField110);
textField111 = new java.awt.TextField();
textField111.setBounds(468,123,25,20);
textField111.setFont(new Font("Dialog", Font.PLAIN, 10));
textField111.setBackground(new Color(-128));
panel4.add(textField111);
textField112 = new java.awt.TextField();
textField112.setBounds(496,123,25,20);
textField112.setFont(new Font("Dialog", Font.PLAIN, 10));
textField112.setBackground(new Color(-8323200));
panel4.add(textField112);
textField113 = new java.awt.TextField();
textField113.setBounds(84,147,45,20);
textField113.setFont(new Font("Dialog", Font.PLAIN, 10));
textField113.setBackground(new Color(-128));
panel4.add(textField113);
textField114 = new java.awt.TextField();
textField114.setBounds(132,147,45,20);
textField114.setFont(new Font("Dialog", Font.PLAIN, 10));
textField114.setBackground(new Color(-8323200));
panel4.add(textField114);
textField115 = new java.awt.TextField();
textField115.setBounds(180,147,45,20);
textField115.setFont(new Font("Dialog", Font.PLAIN, 10));
textField115.setBackground(new Color(-128));
panel4.add(textField115);
textField116 = new java.awt.TextField();
textField116.setBounds(228,147,45,20);
textField116.setFont(new Font("Dialog", Font.PLAIN, 10));
textField116.setBackground(new Color(-8323200));
panel4.add(textField116);
textField117 = new java.awt.TextField();
textField117.setBounds(276,147,45,20);
textField117.setFont(new Font("Dialog", Font.PLAIN, 10));
textField117.setBackground(new Color(-128));
panel4.add(textField117);
textField118 = new java.awt.TextField();
textField118.setBounds(324,147,45,20);
textField118.setFont(new Font("Dialog", Font.PLAIN, 10));
textField118.setBackground(new Color(-8323200));
panel4.add(textField118);
textField119 = new java.awt.TextField();
textField119.setBounds(372,147,45,20);
textField119.setFont(new Font("Dialog", Font.PLAIN, 10));
textField119.setBackground(new Color(-128));
panel4.add(textField119);
textField120 = new java.awt.TextField();
textField120.setBounds(420,147,45,20);
textField120.setFont(new Font("Dialog", Font.PLAIN, 10));
textField120.setBackground(new Color(-8323200));

```

```

panel4.add(textField120);
textField121 = new java.awt.TextField();
textField121.setBounds(468,147,25,20);
textField121.setFont(new Font("Dialog", Font.PLAIN, 10));
textField121.setBackground(new Color(-128));
panel4.add(textField121);
textField122 = new java.awt.TextField();
textField122.setBounds(496,147,25,20);
textField122.setFont(new Font("Dialog", Font.PLAIN, 10));
textField122.setBackground(new Color(-8323200));
panel4.add(textField122);
textField123 = new java.awt.TextField();
textField123.setBounds(84,171,45,20);
textField123.setFont(new Font("Dialog", Font.PLAIN, 10));
textField123.setBackground(new Color(-128));
panel4.add(textField123);
textField124 = new java.awt.TextField();
textField124.setBounds(132,171,45,20);
textField124.setFont(new Font("Dialog", Font.PLAIN, 10));
textField124.setBackground(new Color(-8323200));
panel4.add(textField124);
textField125 = new java.awt.TextField();
textField125.setBounds(180,171,45,20);
textField125.setFont(new Font("Dialog", Font.PLAIN, 10));
textField125.setBackground(new Color(-128));
panel4.add(textField125);
textField126 = new java.awt.TextField();
textField126.setBounds(228,171,45,20);
textField126.setFont(new Font("Dialog", Font.PLAIN, 10));
textField126.setBackground(new Color(-8323200));
panel4.add(textField126);
textField127 = new java.awt.TextField();
textField127.setBounds(276,171,45,20);
textField127.setFont(new Font("Dialog", Font.PLAIN, 10));
textField127.setBackground(new Color(-128));
panel4.add(textField127);
textField128 = new java.awt.TextField();
textField128.setBounds(324,171,45,20);
textField128.setFont(new Font("Dialog", Font.PLAIN, 10));
textField128.setBackground(new Color(-8323200));
panel4.add(textField128);
textField129 = new java.awt.TextField();
textField129.setBounds(372,171,45,20);
textField129.setFont(new Font("Dialog", Font.PLAIN, 10));
textField129.setBackground(new Color(-128));
panel4.add(textField129);
textField130 = new java.awt.TextField();
textField130.setBounds(420,171,45,20);
textField130.setFont(new Font("Dialog", Font.PLAIN, 10));
textField130.setBackground(new Color(-8323200));
panel4.add(textField130);
textField131 = new java.awt.TextField();
textField131.setBounds(468,171,25,20);
textField131.setFont(new Font("Dialog", Font.PLAIN, 10));
textField131.setBackground(new Color(-128));
panel4.add(textField131);

```

```

textField132 = new java.awt.TextField();
textField132.setBounds(496,171,25,20);
textField132.setFont(new Font("Dialog", Font.PLAIN, 10));
textField132.setBackground(new Color(-8323200));
panel4.add(textField132);
textField133 = new java.awt.TextField();
textField133.setBounds(84,195,45,20);
textField133.setFont(new Font("Dialog", Font.PLAIN, 10));
textField133.setBackground(new Color(-128));
panel4.add(textField133);
textField134 = new java.awt.TextField();
textField134.setBounds(132,195,45,20);
textField134.setFont(new Font("Dialog", Font.PLAIN, 10));
textField134.setBackground(new Color(-8323200));
panel4.add(textField134);
textField135 = new java.awt.TextField();
textField135.setBounds(180,195,45,20);
textField135.setFont(new Font("Dialog", Font.PLAIN, 10));
textField135.setBackground(new Color(-128));
panel4.add(textField135);
textField136 = new java.awt.TextField();
textField136.setBounds(228,195,45,20);
textField136.setFont(new Font("Dialog", Font.PLAIN, 10));
textField136.setBackground(new Color(-8323200));
panel4.add(textField136);
textField137 = new java.awt.TextField();
textField137.setBounds(276,195,45,20);
textField137.setFont(new Font("Dialog", Font.PLAIN, 10));
textField137.setBackground(new Color(-128));
panel4.add(textField137);
textField138 = new java.awt.TextField();
textField138.setBounds(324,195,45,20);
textField138.setFont(new Font("Dialog", Font.PLAIN, 10));
textField138.setBackground(new Color(-8323200));
panel4.add(textField138);
textField139 = new java.awt.TextField();
textField139.setBounds(372,195,45,20);
textField139.setFont(new Font("Dialog", Font.PLAIN, 10));
textField139.setBackground(new Color(-128));
panel4.add(textField139);
textField140 = new java.awt.TextField();
textField140.setBounds(420,195,45,20);
textField140.setFont(new Font("Dialog", Font.PLAIN, 10));
textField140.setBackground(new Color(-8323200));
panel4.add(textField140);
textField141 = new java.awt.TextField();
textField141.setBounds(468,195,25,20);
textField141.setFont(new Font("Dialog", Font.PLAIN, 10));
textField141.setBackground(new Color(-128));
panel4.add(textField141);
textField142 = new java.awt.TextField();
textField142.setBounds(496,195,25,20);
textField142.setFont(new Font("Dialog", Font.PLAIN, 10));
textField142.setBackground(new Color(-8323200));
panel4.add(textField142);
label15 = new java.awt.Label("Time Period");

```

```

label15.setBounds(0,51,70,15);
label15.setFont(new Font("Dialog", Font.PLAIN, 11));
panel4.add(label15);
label16 = new java.awt.Label("      #1");
label16.setBounds(84,51,45,15);
label16.setFont(new Font("Dialog", Font.PLAIN, 10));
label16.setBackground(new Color(-128));
panel4.add(label16);
label17 = new java.awt.Label("      #2");
label17.setBounds(132,51,45,15);
label17.setFont(new Font("Dialog", Font.PLAIN, 10));
label17.setBackground(new Color(-8323200));
panel4.add(label17);
label18 = new java.awt.Label("      Volts");
label18.setBounds(180,51,45,15);
label18.setFont(new Font("Dialog", Font.PLAIN, 9));
label18.setBackground(new Color(-128));
panel4.add(label18);
label19 = new java.awt.Label("      AB");
label19.setBounds(228,51,45,15);
label19.setFont(new Font("Dialog", Font.PLAIN, 10));
label19.setBackground(new Color(-8323200));
panel4.add(label19);
label20 = new java.awt.Label("      BC");
label20.setBounds(276,51,45,15);
label20.setFont(new Font("Dialog", Font.PLAIN, 10));
label20.setBackground(new Color(-128));
panel4.add(label20);
label21 = new java.awt.Label("      AC");
label21.setBounds(324,51,45,15);
label21.setFont(new Font("Dialog", Font.PLAIN, 10));
label21.setBackground(new Color(-8323200));
panel4.add(label21);
label22 = new java.awt.Label("Agitation");
label22.setBounds(372,51,45,15);
label22.setFont(new Font("Dialog", Font.PLAIN, 10));
label22.setBackground(new Color(-128));
panel4.add(label22);
label23 = new java.awt.Label("Filt DP");
label23.setBounds(420,51,45,15);
label23.setFont(new Font("Dialog", Font.PLAIN, 9));
label23.setBackground(new Color(-8323200));
panel4.add(label23);
label24 = new java.awt.Label("Fuse");
label24.setBounds(468,51,25,15);
label24.setFont(new Font("Dialog", Font.PLAIN, 10));
label24.setBackground(new Color(-128));
panel4.add(label24);
label25 = new java.awt.Label("Lmps");
label25.setBounds(496,51,25,15);
label25.setFont(new Font("Dialog", Font.PLAIN, 10));
label25.setBackground(new Color(-8323200));
panel4.add(label25);
textField143 = new java.awt.TextField();
textField143.setBounds(524,75,25,20);
textField143.setFont(new Font("Dialog", Font.PLAIN, 10));

```

```

textField143.setBackground(new Color(-128));
panel4.add(textField143);
textField144 = new java.awt.TextField();
textField144.setBounds(524,99,25,20);
textField144.setFont(new Font("Dialog", Font.PLAIN, 10));
textField144.setBackground(new Color(-128));
panel4.add(textField144);
textField145 = new java.awt.TextField();
textField145.setBounds(524,123,25,20);
textField145.setFont(new Font("Dialog", Font.PLAIN, 10));
textField145.setBackground(new Color(-128));
panel4.add(textField145);
textField146 = new java.awt.TextField();
textField146.setBounds(524,147,25,20);
textField146.setFont(new Font("Dialog", Font.PLAIN, 10));
textField146.setBackground(new Color(-128));
panel4.add(textField146);
textField147 = new java.awt.TextField();
textField147.setBounds(524,171,25,20);
textField147.setFont(new Font("Dialog", Font.PLAIN, 10));
textField147.setBackground(new Color(-128));
panel4.add(textField147);
textField148 = new java.awt.TextField();
textField148.setBounds(524,195,25,20);
textField148.setFont(new Font("Dialog", Font.PLAIN, 10));
textField148.setBackground(new Color(-128));
panel4.add(textField148);
label26 = new java.awt.Label(" 19");
label26.setBounds(524,51,25,15);
label26.setFont(new Font("Dialog", Font.PLAIN, 10));
label26.setBackground(new Color(-128));
panel4.add(label26);
textField149 = new java.awt.TextField();
textField149.setBounds(552,75,25,20);
textField149.setFont(new Font("Dialog", Font.PLAIN, 10));
textField149.setBackground(new Color(-8323200));
panel4.add(textField149);
textField150 = new java.awt.TextField();
textField150.setBounds(552,99,25,20);
textField150.setFont(new Font("Dialog", Font.PLAIN, 10));
textField150.setBackground(new Color(-8323200));
panel4.add(textField150);
textField151 = new java.awt.TextField();
textField151.setBounds(552,123,25,20);
textField151.setFont(new Font("Dialog", Font.PLAIN, 10));
textField151.setBackground(new Color(-8323200));
panel4.add(textField151);
textField152 = new java.awt.TextField();
textField152.setBounds(552,147,25,20);
textField152.setFont(new Font("Dialog", Font.PLAIN, 10));
textField152.setBackground(new Color(-8323200));
panel4.add(textField152);
textField153 = new java.awt.TextField();
textField153.setBounds(552,171,25,20);
textField153.setFont(new Font("Dialog", Font.PLAIN, 10));
textField153.setBackground(new Color(-8323200));

```

```

panel4.add(textField153);
textField154 = new java.awt.TextField();
textField154.setBounds(552,195,25,20);
textField154.setFont(new Font("Dialog", Font.PLAIN, 10));
textField154.setBackground(new Color(-8323200));
panel4.add(textField154);
label27 = new java.awt.Label(" Det");
label27.setBounds(552,51,36,15);
label27.setFont(new Font("Dialog", Font.PLAIN, 10));
label27.setBackground(new Color(-8323200));
panel4.add(label27);
label28 = new java.awt.Label("H2 Detect");
label28.setBounds(84,39,45,15);
label28.setFont(new Font("Dialog", Font.PLAIN, 10));
label28.setBackground(new Color(-128));
panel4.add(label28);
label29 = new java.awt.Label("H2 Detect");
label29.setBounds(132,39,45,15);
label29.setFont(new Font("Dialog", Font.PLAIN, 10));
label29.setBackground(new Color(-8323200));
panel4.add(label29);
label30 = new java.awt.Label("AN / WIC");
label30.setBounds(180,39,45,15);
label30.setFont(new Font("Dialog", Font.PLAIN, 9));
label30.setBackground(new Color(-128));
panel4.add(label30);
label34 = new java.awt.Label("Battery");
label34.setBounds(372,39,45,15);
label34.setFont(new Font("Dialog", Font.PLAIN, 10));
label34.setBackground(new Color(-128));
panel4.add(label34);
label35 = new java.awt.Label("Battery");
label35.setBounds(420,39,45,15);
label35.setFont(new Font("Dialog", Font.PLAIN, 9));
label35.setBackground(new Color(-8323200));
panel4.add(label35);
label36 = new java.awt.Label("IC");
label36.setBounds(468,39,25,15);
label36.setFont(new Font("Dialog", Font.PLAIN, 10));
label36.setBackground(new Color(-128));
panel4.add(label36);
label37 = new java.awt.Label("Sync");
label37.setBounds(496,39,25,15);
label37.setFont(new Font("Dialog", Font.PLAIN, 10));
label37.setBackground(new Color(-8323200));
panel4.add(label37);
label38 = new java.awt.Label(" MK");
label38.setBounds(524,39,25,15);
label38.setFont(new Font("Dialog", Font.PLAIN, 10));
label38.setBackground(new Color(-128));
panel4.add(label38);
label39 = new java.awt.Label("EIC Gnd");
label39.setBounds(552,39,36,15);
label39.setFont(new Font("Dialog", Font.PLAIN, 10));
label39.setBackground(new Color(-8323200));
panel4.add(label39);

```

```

label31 = new java.awt.Label("      Static inverter
(Volts)");
label31.setBounds(228,36,141,15);
label31.setBackground(new Color(-128));
panel4.add(label31);
textField155 = new java.awt.TextField();
textField155.setBounds(84,255,45,20);
textField155.setFont(new Font("Dialog", Font.PLAIN, 10));
textField155.setBackground(new Color(16777215));
panel4.add(textField155);
textField156 = new java.awt.TextField();
textField156.setBounds(132,255,45,20);
textField156.setFont(new Font("Dialog", Font.PLAIN, 10));
textField156.setBackground(new Color(16777215));
panel4.add(textField156);
textField157 = new java.awt.TextField();
textField157.setBounds(180,255,45,20);
textField157.setFont(new Font("Dialog", Font.PLAIN, 10));
textField157.setBackground(new Color(16777215));
panel4.add(textField157);
textField158 = new java.awt.TextField();
textField158.setBounds(228,255,45,20);
textField158.setFont(new Font("Dialog", Font.PLAIN, 10));
textField158.setBackground(new Color(16777215));
panel4.add(textField158);
textField159 = new java.awt.TextField();
textField159.setBounds(276,255,45,20);
textField159.setFont(new Font("Dialog", Font.PLAIN, 10));
textField159.setBackground(new Color(16777215));
panel4.add(textField159);
textField160 = new java.awt.TextField();
textField160.setBounds(324,255,45,20);
textField160.setFont(new Font("Dialog", Font.PLAIN, 10));
textField160.setBackground(new Color(16777215));
panel4.add(textField160);
textField161 = new java.awt.TextField();
textField161.setBounds(372,255,45,20);
textField161.setFont(new Font("Dialog", Font.PLAIN, 10));
textField161.setBackground(new Color(16777215));
panel4.add(textField161);
textField162 = new java.awt.TextField();
textField162.setBounds(420,255,45,20);
textField162.setFont(new Font("Dialog", Font.PLAIN, 10));
textField162.setBackground(new Color(16777215));
panel4.add(textField162);
textField163 = new java.awt.TextField();
textField163.setBounds(468,255,25,20);
textField163.setFont(new Font("Dialog", Font.PLAIN, 10));
textField163.setBackground(new Color(16777215));
panel4.add(textField163);
textField164 = new java.awt.TextField();
textField164.setBounds(496,255,25,20);
textField164.setFont(new Font("Dialog", Font.PLAIN, 10));
textField164.setBackground(new Color(16777215));
panel4.add(textField164);
textField165 = new java.awt.TextField();

```

```

textField165.setBounds(524,255,25,20);
textField165.setFont(new Font("Dialog", Font.PLAIN, 10));
textField165.setBackground(new Color(16777215));
panel4.add(textField165);
textField166 = new java.awt.TextField();
textField166.setBounds(552,255,25,20);
textField166.setFont(new Font("Dialog", Font.PLAIN, 10));
textField166.setBackground(new Color(16777215));
panel4.add(textField166);
button4 = new java.awt.Button();
button4.addActionListener(this);
button4.setLabel("Submit");
button4.setBounds(504,303,68,19);
button4.setBackground(new Color(12632256));
panel4.add(button4);
button5 = new java.awt.Button();
button5.addActionListener(this);
button5.setLabel("Retrieve");
button5.setBounds(420,303,69,19);
button5.setBackground(new Color(12632256));
panel4.add(button5);
choice2 = new java.awt.Choice();
choice2.addItem("00:00");
choice2.addItem("01:00");
choice2.addItem("02:00");
choice2.addItem("03:00");
choice2.addItem("04:00");
choice2.addItem("05:00");
choice2.addItem("06:00");
choice2.addItem("07:00");
choice2.addItem("08:00");
choice2.addItem("09:00");
choice2.addItem("10:00");
choice2.addItem("11:00");
choice2.addItem("12:00");
choice2.addItem("13:00");
choice2.addItem("14:00");
choice2.addItem("15:00");
choice2.addItem("16:00");
choice2.addItem("17:00");
choice2.addItem("18:00");
choice2.addItem("19:00");
choice2.addItem("20:00");
choice2.addItem("21:00");
choice2.addItem("22:00");
choice2.addItem("23:00");
panel4.add(choice2);
choice2.setBounds(0,255,67,20);
choice2.setBackground(new Color(16777215));
panel2 = new java.awt.Panel();
panel2.setLayout(null);
panel2.setVisible(false);
panel2.setBounds(12,33,636,436);
tabPanel1.add(panel2);
panel5 = new java.awt.Panel();
panel5.setLayout(null);

```

```

panel5.setBounds(12,27,611,380);
panel5.setBackground(new Color(12632256));
panel2.add(panel5);
textField167 = new java.awt.TextField();
textField167.setEditable(false);
textField167.setBounds(48,120,67,20);
textField167.setFont(new Font("Dialog", Font.PLAIN, 10));
textField167.setForeground(new Color(0));
textField167.setBackground(new Color(16777215));
panel5.add(textField167);
textField168 = new java.awt.TextField();
textField168.setBounds(48,144,67,20);
textField168.setFont(new Font("Dialog", Font.PLAIN, 10));
textField168.setBackground(new Color(16777215));
panel5.add(textField168);
textField169 = new java.awt.TextField();
textField169.setBounds(48,168,67,20);
textField169.setFont(new Font("Dialog", Font.PLAIN, 10));
textField169.setBackground(new Color(16777215));
panel5.add(textField169);
textField170 = new java.awt.TextField();
textField170.setBounds(48,192,67,20);
textField170.setFont(new Font("Dialog", Font.PLAIN, 10));
textField170.setBackground(new Color(16777215));
panel5.add(textField170);
textField171 = new java.awt.TextField();
textField171.setBounds(48,216,67,20);
textField171.setFont(new Font("Dialog", Font.PLAIN, 10));
textField171.setBackground(new Color(16777215));
panel5.add(textField171);
textField172 = new java.awt.TextField();
textField172.setBounds(48,240,67,20);
textField172.setFont(new Font("Dialog", Font.PLAIN, 10));
textField172.setBackground(new Color(16777215));
panel5.add(textField172);
textField173 = new java.awt.TextField();
textField173.setBounds(132,120,65,20);
textField173.setFont(new Font("Dialog", Font.PLAIN, 10));
textField173.setBackground(new Color(-128));
panel5.add(textField173);
textField174 = new java.awt.TextField();
textField174.setBounds(132,144,65,20);
textField174.setFont(new Font("Dialog", Font.PLAIN, 10));
textField174.setBackground(new Color(-128));
panel5.add(textField174);
textField175 = new java.awt.TextField();
textField175.setBounds(132,168,65,20);
textField175.setFont(new Font("Dialog", Font.PLAIN, 10));
textField175.setBackground(new Color(-128));
panel5.add(textField175);
textField176 = new java.awt.TextField();
textField176.setBounds(132,192,65,20);
textField176.setFont(new Font("Dialog", Font.PLAIN, 10));
textField176.setBackground(new Color(-128));
panel5.add(textField176);
textField177 = new java.awt.TextField();

```

```

textField177.setBounds(132,216,65,20);
textField177.setFont(new Font("Dialog", Font.PLAIN, 10));
textField177.setBackground(new Color(-128));
panel5.add(textField177);
textField178 = new java.awt.TextField();
textField178.setBounds(132,240,65,20);
textField178.setFont(new Font("Dialog", Font.PLAIN, 10));
textField178.setBackground(new Color(-128));
panel5.add(textField178);
label33 = new java.awt.Label("Time Period");
label33.setBounds(48,96,67,15);
panel5.add(label33);
label40 = new java.awt.Label("CO");
label40.setBounds(132,96,65,15);
label40.setFont(new Font("Dialog", Font.PLAIN, 10));
label40.setBackground(new Color(-128));
panel5.add(label40);
textField179 = new java.awt.TextField();
textField179.setBounds(204,120,65,20);
textField179.setFont(new Font("Dialog", Font.PLAIN, 10));
textField179.setBackground(new Color(-8323200));
panel5.add(textField179);
textField180 = new java.awt.TextField();
textField180.setBounds(204,144,65,20);
textField180.setFont(new Font("Dialog", Font.PLAIN, 10));
textField180.setBackground(new Color(-8323200));
panel5.add(textField180);
textField181 = new java.awt.TextField();
textField181.setBounds(204,168,65,20);
textField181.setFont(new Font("Dialog", Font.PLAIN, 10));
textField181.setBackground(new Color(-8323200));
panel5.add(textField181);
textField182 = new java.awt.TextField();
textField182.setBounds(204,192,65,20);
textField182.setFont(new Font("Dialog", Font.PLAIN, 10));
textField182.setBackground(new Color(-8323200));
panel5.add(textField182);
textField183 = new java.awt.TextField();
textField183.setBounds(204,216,65,20);
textField183.setFont(new Font("Dialog", Font.PLAIN, 10));
textField183.setBackground(new Color(-8323200));
panel5.add(textField183);
textField184 = new java.awt.TextField();
textField184.setBounds(204,240,65,20);
textField184.setFont(new Font("Dialog", Font.PLAIN, 10));
textField184.setBackground(new Color(-8323200));
panel5.add(textField184);
label41 = new java.awt.Label("R-114");
label41.setBounds(204,96,65,15);
label41.setFont(new Font("Dialog", Font.PLAIN, 10));
label41.setBackground(new Color(-8323200));
panel5.add(label41);
textField185 = new java.awt.TextField();
textField185.setBounds(276,120,65,20);
textField185.setFont(new Font("Dialog", Font.PLAIN, 10));
textField185.setBackground(new Color(-128));

```

```

panel5.add(textField185);
textField186 = new java.awt.TextField();
textField186.setBounds(276,144,65,20);
textField186.setFont(new Font("Dialog", Font.PLAIN, 10));
textField186.setBackground(new Color(-128));
panel5.add(textField186);
textField187 = new java.awt.TextField();
textField187.setBounds(276,168,65,20);
textField187.setFont(new Font("Dialog", Font.PLAIN, 10));
textField187.setBackground(new Color(-128));
panel5.add(textField187);
textField188 = new java.awt.TextField();
textField188.setBounds(276,192,65,20);
textField188.setFont(new Font("Dialog", Font.PLAIN, 10));
textField188.setBackground(new Color(-128));
panel5.add(textField188);
textField189 = new java.awt.TextField();
textField189.setBounds(276,216,65,20);
textField189.setFont(new Font("Dialog", Font.PLAIN, 10));
textField189.setBackground(new Color(-128));
panel5.add(textField189);
textField190 = new java.awt.TextField();
textField190.setBounds(276,240,65,20);
textField190.setFont(new Font("Dialog", Font.PLAIN, 10));
textField190.setBackground(new Color(-128));
panel5.add(textField190);
label42 = new java.awt.Label("      R-12");
label42.setBounds(276,96,65,15);
label42.setFont(new Font("Dialog", Font.PLAIN, 10));
label42.setBackground(new Color(-128));
panel5.add(label42);
textField191 = new java.awt.TextField();
textField191.setBounds(348,120,65,20);
textField191.setFont(new Font("Dialog", Font.PLAIN, 10));
textField191.setBackground(new Color(-8323200));
panel5.add(textField191);
textField192 = new java.awt.TextField();
textField192.setBounds(348,144,65,20);
textField192.setFont(new Font("Dialog", Font.PLAIN, 10));
textField192.setBackground(new Color(-8323200));
panel5.add(textField192);
textField193 = new java.awt.TextField();
textField193.setBounds(348,168,65,20);
textField193.setFont(new Font("Dialog", Font.PLAIN, 10));
textField193.setBackground(new Color(-8323200));
panel5.add(textField193);
textField194 = new java.awt.TextField();
textField194.setBounds(348,192,65,20);
textField194.setFont(new Font("Dialog", Font.PLAIN, 10));
textField194.setBackground(new Color(-8323200));
panel5.add(textField194);
textField195 = new java.awt.TextField();
textField195.setBounds(348,216,65,20);
textField195.setFont(new Font("Dialog", Font.PLAIN, 10));
textField195.setBackground(new Color(-8323200));
panel5.add(textField195);

```

```

textField196 = new java.awt.TextField();
textField196.setBounds(348,240,65,20);
textField196.setFont(new Font("Dialog", Font.PLAIN, 10));
textField196.setBackground(new Color(-8323200));
panel5.add(textField196);
label43 = new java.awt.Label("      O2");
label43.setBounds(348,96,65,15);
label43.setFont(new Font("Dialog", Font.PLAIN, 10));
label43.setBackground(new Color(-8323200));
panel5.add(label43);
textField197 = new java.awt.TextField();
textField197.setBounds(420,120,65,20);
textField197.setFont(new Font("Dialog", Font.PLAIN, 10));
textField197.setBackground(new Color(-128));
panel5.add(textField197);
textField198 = new java.awt.TextField();
textField198.setBounds(420,144,65,20);
textField198.setFont(new Font("Dialog", Font.PLAIN, 10));
textField198.setBackground(new Color(-128));
panel5.add(textField198);
textField199 = new java.awt.TextField();
textField199.setBounds(420,168,65,20);
textField199.setFont(new Font("Dialog", Font.PLAIN, 10));
textField199.setBackground(new Color(-128));
panel5.add(textField199);
textField200 = new java.awt.TextField();
textField200.setBounds(420,192,65,20);
textField200.setFont(new Font("Dialog", Font.PLAIN, 10));
textField200.setBackground(new Color(-128));
panel5.add(textField200);
textField201 = new java.awt.TextField();
textField201.setBounds(420,216,65,20);
textField201.setFont(new Font("Dialog", Font.PLAIN, 10));
textField201.setBackground(new Color(-128));
panel5.add(textField201);
textField202 = new java.awt.TextField();
textField202.setBounds(420,240,65,20);
textField202.setFont(new Font("Dialog", Font.PLAIN, 10));
textField202.setBackground(new Color(-128));
panel5.add(textField202);
label44 = new java.awt.Label("      CO2");
label44.setBounds(420,96,65,15);
label44.setFont(new Font("Dialog", Font.PLAIN, 10));
label44.setBackground(new Color(-128));
panel5.add(label44);
textField203 = new java.awt.TextField();
textField203.setBounds(492,120,65,20);
textField203.setFont(new Font("Dialog", Font.PLAIN, 10));
textField203.setBackground(new Color(-8323200));
panel5.add(textField203);
textField204 = new java.awt.TextField();
textField204.setBounds(492,144,65,20);
textField204.setFont(new Font("Dialog", Font.PLAIN, 10));
textField204.setBackground(new Color(-8323200));
panel5.add(textField204);
textField205 = new java.awt.TextField();

```

```

textField205.setBounds(492,168,65,20);
textField205.setFont(new Font("Dialog", Font.PLAIN, 10));
textField205.setBackground(new Color(-8323200));
panel5.add(textField205);
textField206 = new java.awt.TextField();
textField206.setBounds(492,192,65,20);
textField206.setFont(new Font("Dialog", Font.PLAIN, 10));
textField206.setBackground(new Color(-8323200));
panel5.add(textField206);
textField207 = new java.awt.TextField();
textField207.setBounds(492,216,65,20);
textField207.setFont(new Font("Dialog", Font.PLAIN, 10));
textField207.setBackground(new Color(-8323200));
panel5.add(textField207);
textField208 = new java.awt.TextField();
textField208.setBounds(492,240,65,20);
textField208.setFont(new Font("Dialog", Font.PLAIN, 10));
textField208.setBackground(new Color(-8323200));
panel5.add(textField208);
label45 = new java.awt.Label("          H2");
label45.setBounds(492,96,65,15);
label45.setFont(new Font("Dialog", Font.PLAIN, 10));
label45.setBackground(new Color(-8323200));
panel5.add(label45);
label46 = new java.awt.Label("Total Pressure :");
label46.setBounds(48,324,96,20);
label46.setFont(new Font("Dialog", Font.BOLD, 12));
panel5.add(label46);
textField209 = new java.awt.TextField();
textField209.setBounds(168,324,65,20);
panel5.add(textField209);
textField210 = new java.awt.TextField();
textField210.setBounds(132,288,65,20);
textField210.setFont(new Font("Dialog", Font.PLAIN, 10));
textField210.setBackground(new Color(16777215));
panel5.add(textField210);
textField211 = new java.awt.TextField();
textField211.setBounds(204,288,65,20);
textField211.setFont(new Font("Dialog", Font.PLAIN, 10));
textField211.setBackground(new Color(16777215));
panel5.add(textField211);
textField212 = new java.awt.TextField();
textField212.setBounds(276,288,65,20);
textField212.setFont(new Font("Dialog", Font.PLAIN, 10));
textField212.setBackground(new Color(16777215));
panel5.add(textField212);
textField213 = new java.awt.TextField();
textField213.setBounds(348,288,65,20);
textField213.setFont(new Font("Dialog", Font.PLAIN, 10));
textField213.setBackground(new Color(16777215));
panel5.add(textField213);
textField214 = new java.awt.TextField();
textField214.setBounds(420,288,65,20);
textField214.setFont(new Font("Dialog", Font.PLAIN, 10));
textField214.setBackground(new Color(16777215));
panel5.add(textField214);

```

```

textField215 = new java.awt.TextField();
textField215.setBounds(492,288,65,20);
textField215.setFont(new Font("Dialog", Font.PLAIN, 10));
textField215.setBackground(new Color(16777215));
panel5.add(textField215);
label47 = new java.awt.Label(" POSITION  :");
label47.setBounds(48,48,72,20);
label47.setFont(new Font("Dialog", Font.BOLD, 12));
panel5.add(label47);
button6 = new java.awt.Button();
button6.addActionListener(this);
button6.setLabel("Submit");
button6.setBounds(528,336,68,19);
button6.setBackground(new Color(12632256));
panel5.add(button6);
button7 = new java.awt.Button();
button7.addActionListener(this);
button7.setLabel("Retrieve");
button7.setBounds(444,336,69,19);
button7.setBackground(new Color(12632256));
panel5.add(button7);
choice3 = new java.awt.Choice();
choice3.addItem("00:00");
choice3.addItem("01:00");
choice3.addItem("02:00");
choice3.addItem("03:00");
choice3.addItem("04:00");
choice3.addItem("05:00");
choice3.addItem("06:00");
choice3.addItem("07:00");
choice3.addItem("08:00");
choice3.addItem("09:00");
choice3.addItem("10:00");
choice3.addItem("11:00");
choice3.addItem("12:00");
choice3.addItem("13:00");
choice3.addItem("14:00");
choice3.addItem("15:00");
choice3.addItem("16:00");
choice3.addItem("17:00");
choice3.addItem("18:00");
choice3.addItem("19:00");
choice3.addItem("20:00");
choice3.addItem("21:00");
choice3.addItem("22:00");
choice3.addItem("23:00");
panel5.add(choice3);
choice3.setBounds(48,288,67,20);
choice3.setBackground(new Color(16777215));
choice5 = new java.awt.Choice();
choice5.addItem("1");
choice5.addItem("2");
choice5.addItem("3");
choice5.addItem("4");
choice5.addItem("5");
choice5.addItem("6");

```

```

choice5.addItem("7");
choice5.addItem("8");
panel5.add(choice5);
choice5.setBounds(132,48,65,20);
choice5.setBackground(new Color(16777215));
panel6 = new java.awt.Panel();
panel6.setLayout(null);
panel6.setVisible(false);
panel6.setBounds(12,33,636,436);
tabPanel1.add(panel6);
textArea1 = new java.awt.TextArea();
textArea1.setBounds(24,75,588,132);
textArea1.setBackground(new Color(16777215));
panel6.add(textArea1);
button3 = new java.awt.Button();
button3.addActionListener(this);
button3.setLabel("Submit");
button3.setBounds(516,387,86,23);
button3.setBackground(new Color(12632256));
panel6.add(button3);
label49 = new java.awt.Label("Remarks :");
label49.setBounds(24,39,104,19);
label49.setFont(new Font("Dialog", Font.BOLD, 12));
panel6.add(label49);
choice4 = new java.awt.Choice();
choice4.addItem("00:00");
choice4.addItem("01:00");
choice4.addItem("02:00");
choice4.addItem("03:00");
choice4.addItem("04:00");
choice4.addItem("05:00");
choice4.addItem("06:00");
choice4.addItem("07:00");
choice4.addItem("08:00");
choice4.addItem("09:00");
choice4.addItem("10:00");
choice4.addItem("11:00");
choice4.addItem("12:00");
choice4.addItem("13:00");
choice4.addItem("14:00");
choice4.addItem("15:00");
choice4.addItem("16:00");
choice4.addItem("17:00");
choice4.addItem("18:00");
choice4.addItem("19:00");
choice4.addItem("20:00");
choice4.addItem("21:00");
choice4.addItem("22:00");
choice4.addItem("23:00");
panel6.add(choice4);
choice4.setBounds(24,351,67,20);
choice4.setBackground(new Color(16777215));
button8 = new java.awt.Button();
button8.addActionListener(this);
button8.setLabel("Receive");
button8.setBounds(410,387,86,23);

```

```

        button8.setBackground(new Color(12632256));
        panel6.add(button8);
        textArea2 = new java.awt.TextArea();
        textArea2.setBounds(24, 219, 588, 124);
        panel6.add(textArea2);
        //}}
    }

```

```

//{{DECLARE_CONTROLS
symantec.itools.awt.TabPanel tabPanel1;
java.awt.Panel panel1;
symantec.itools.awt.TabPanel tabPanel2;
java.awt.Panel panel3;
java.awt.Button button1;
java.awt.Button button2;
java.awt.TextField textField1;
java.awt.TextField textField2;
java.awt.TextField textField3;
java.awt.TextField textField4;
java.awt.TextField textField5;
java.awt.TextField textField6;
java.awt.TextField textField7;
java.awt.TextField textField8;
java.awt.TextField textField9;
java.awt.TextField textField10;
java.awt.TextField textField11;
java.awt.TextField textField12;
java.awt.TextField textField13;
java.awt.TextField textField14;
java.awt.TextField textField15;
java.awt.TextField textField16;
java.awt.TextField textField17;
java.awt.TextField textField18;
java.awt.TextField textField19;
java.awt.TextField textField20;
java.awt.TextField textField21;
java.awt.TextField textField22;
java.awt.TextField textField23;
java.awt.TextField textField24;
java.awt.TextField textField25;
java.awt.TextField textField26;
java.awt.TextField textField27;
java.awt.TextField textField28;
java.awt.TextField textField29;
java.awt.TextField textField30;
java.awt.TextField textField31;
java.awt.TextField textField32;
java.awt.TextField textField33;
java.awt.TextField textField34;
java.awt.TextField textField35;
java.awt.TextField textField36;
java.awt.TextField textField37;
java.awt.TextField textField38;
java.awt.TextField textField39;
java.awt.TextField textField40;
java.awt.TextField textField41;

```

```
java.awt.TextField textField42;  
java.awt.TextField textField43;  
java.awt.TextField textField44;  
java.awt.TextField textField45;  
java.awt.TextField textField46;  
java.awt.TextField textField47;  
java.awt.TextField textField48;  
java.awt.TextField textField49;  
java.awt.TextField textField50;  
java.awt.TextField textField51;  
java.awt.TextField textField52;  
java.awt.TextField textField53;  
java.awt.TextField textField54;  
java.awt.TextField textField55;  
java.awt.TextField textField56;  
java.awt.TextField textField57;  
java.awt.TextField textField58;  
java.awt.TextField textField59;  
java.awt.TextField textField60;  
java.awt.TextField textField61;  
java.awt.TextField textField62;  
java.awt.TextField textField63;  
java.awt.TextField textField64;  
java.awt.TextField textField65;  
java.awt.TextField textField66;  
java.awt.Label label1;  
java.awt.Label label2;  
java.awt.Label label3;  
java.awt.Label label4;  
java.awt.Label label5;  
java.awt.Label label6;  
java.awt.Label label7;  
java.awt.Label label8;  
java.awt.Label label9;  
java.awt.Label label10;  
java.awt.Label label11;  
java.awt.Label label12;  
java.awt.Label label13;  
java.awt.TextField textField67;  
java.awt.TextField textField68;  
java.awt.TextField textField69;  
java.awt.TextField textField70;  
java.awt.TextField textField71;  
java.awt.TextField textField72;  
java.awt.TextField textField73;  
java.awt.TextField textField74;  
java.awt.TextField textField75;  
java.awt.TextField textField76;  
java.awt.Choice choice1;  
java.awt.Panel panel4;  
java.awt.TextField textField77;  
java.awt.TextField textField78;  
java.awt.TextField textField79;  
java.awt.TextField textField80;  
java.awt.TextField textField81;  
java.awt.TextField textField82;
```

```
java.awt.TextField textField83;  
java.awt.TextField textField84;  
java.awt.TextField textField85;  
java.awt.TextField textField86;  
java.awt.TextField textField87;  
java.awt.TextField textField88;  
java.awt.TextField textField89;  
java.awt.TextField textField90;  
java.awt.TextField textField91;  
java.awt.TextField textField92;  
java.awt.TextField textField93;  
java.awt.TextField textField94;  
java.awt.TextField textField95;  
java.awt.TextField textField96;  
java.awt.TextField textField97;  
java.awt.TextField textField98;  
java.awt.TextField textField99;  
java.awt.TextField textField100;  
java.awt.TextField textField101;  
java.awt.TextField textField102;  
java.awt.TextField textField103;  
java.awt.TextField textField104;  
java.awt.TextField textField105;  
java.awt.TextField textField106;  
java.awt.TextField textField107;  
java.awt.TextField textField108;  
java.awt.TextField textField109;  
java.awt.TextField textField110;  
java.awt.TextField textField111;  
java.awt.TextField textField112;  
java.awt.TextField textField113;  
java.awt.TextField textField114;  
java.awt.TextField textField115;  
java.awt.TextField textField116;  
java.awt.TextField textField117;  
java.awt.TextField textField118;  
java.awt.TextField textField119;  
java.awt.TextField textField120;  
java.awt.TextField textField121;  
java.awt.TextField textField122;  
java.awt.TextField textField123;  
java.awt.TextField textField124;  
java.awt.TextField textField125;  
java.awt.TextField textField126;  
java.awt.TextField textField127;  
java.awt.TextField textField128;  
java.awt.TextField textField129;  
java.awt.TextField textField130;  
java.awt.TextField textField131;  
java.awt.TextField textField132;  
java.awt.TextField textField133;  
java.awt.TextField textField134;  
java.awt.TextField textField135;  
java.awt.TextField textField136;  
java.awt.TextField textField137;  
java.awt.TextField textField138;
```

```
java.awt.TextField textField139;  
java.awt.TextField textField140;  
java.awt.TextField textField141;  
java.awt.TextField textField142;  
java.awt.Label label15;  
java.awt.Label label16;  
java.awt.Label label17;  
java.awt.Label label18;  
java.awt.Label label19;  
java.awt.Label label20;  
java.awt.Label label21;  
java.awt.Label label22;  
java.awt.Label label23;  
java.awt.Label label24;  
java.awt.Label label25;  
java.awt.TextField textField143;  
java.awt.TextField textField144;  
java.awt.TextField textField145;  
java.awt.TextField textField146;  
java.awt.TextField textField147;  
java.awt.TextField textField148;  
java.awt.Label label26;  
java.awt.TextField textField149;  
java.awt.TextField textField150;  
java.awt.TextField textField151;  
java.awt.TextField textField152;  
java.awt.TextField textField153;  
java.awt.TextField textField154;  
java.awt.Label label27;  
java.awt.Label label28;  
java.awt.Label label29;  
java.awt.Label label30;  
java.awt.Label label34;  
java.awt.Label label35;  
java.awt.Label label36;  
java.awt.Label label37;  
java.awt.Label label38;  
java.awt.Label label39;  
java.awt.Label label31;  
java.awt.TextField textField155;  
java.awt.TextField textField156;  
java.awt.TextField textField157;  
java.awt.TextField textField158;  
java.awt.TextField textField159;  
java.awt.TextField textField160;  
java.awt.TextField textField161;  
java.awt.TextField textField162;  
java.awt.TextField textField163;  
java.awt.TextField textField164;  
java.awt.TextField textField165;  
java.awt.TextField textField166;  
java.awt.Button button4;  
java.awt.Button button5;  
java.awt.Choice choice2;  
java.awt.Panel panel2;  
java.awt.Panel panel5;
```

```
java.awt.TextField textField167;  
java.awt.TextField textField168;  
java.awt.TextField textField169;  
java.awt.TextField textField170;  
java.awt.TextField textField171;  
java.awt.TextField textField172;  
java.awt.TextField textField173;  
java.awt.TextField textField174;  
java.awt.TextField textField175;  
java.awt.TextField textField176;  
java.awt.TextField textField177;  
java.awt.TextField textField178;  
java.awt.Label label133;  
java.awt.Label label140;  
java.awt.TextField textField179;  
java.awt.TextField textField180;  
java.awt.TextField textField181;  
java.awt.TextField textField182;  
java.awt.TextField textField183;  
java.awt.TextField textField184;  
java.awt.Label label141;  
java.awt.TextField textField185;  
java.awt.TextField textField186;  
java.awt.TextField textField187;  
java.awt.TextField textField188;  
java.awt.TextField textField189;  
java.awt.TextField textField190;  
java.awt.Label label142;  
java.awt.TextField textField191;  
java.awt.TextField textField192;  
java.awt.TextField textField193;  
java.awt.TextField textField194;  
java.awt.TextField textField195;  
java.awt.TextField textField196;  
java.awt.Label label143;  
java.awt.TextField textField197;  
java.awt.TextField textField198;  
java.awt.TextField textField199;  
java.awt.TextField textField200;  
java.awt.TextField textField201;  
java.awt.TextField textField202;  
java.awt.Label label144;  
java.awt.TextField textField203;  
java.awt.TextField textField204;  
java.awt.TextField textField205;  
java.awt.TextField textField206;  
java.awt.TextField textField207;  
java.awt.TextField textField208;  
java.awt.Label label145;  
java.awt.Label label146;  
java.awt.TextField textField209;  
java.awt.TextField textField210;  
java.awt.TextField textField211;  
java.awt.TextField textField212;  
java.awt.TextField textField213;  
java.awt.TextField textField214;
```

```

java.awt.TextField textField215;
java.awt.Label label47;
java.awt.Button button6;
java.awt.Button button7;
java.awt.Choice choice3;
java.awt.Choice choice5;
java.awt.Panel panel6;
java.awt.TextArea textArea1;
java.awt.Button button3;
java.awt.Label label49;
java.awt.Choice choice4;
java.awt.Button button8;
java.awt.TextArea textArea2;
//}}

// DATABASE VARIABLES

TextField theStatus      = new TextField(64);
JDBC01 theJDBC = new JDBC01(theStatus);

public void actionPerformed (ActionEvent event)
{
    Object source = event.getSource();

    // 400 cycle set tab
    if (source == button1) {

        boolean temp = true;

        String t1 = textField67.getText();
        String t2 = textField68.getText();
        String t3 = textField69.getText();
        String t4 = textField70.getText();
        String t5 = textField71.getText();
        String t6 = textField72.getText();
        String t7 = textField73.getText();
        String t8 = textField74.getText();
        String t9 = textField75.getText();
        String t10 = textField76.getText();

        if ( t1.equals(""))
            t1 = "N/A";
        if ( t2.equals(""))
            t2 = "N/A";
        if ( t3.equals(""))
            t3 = "N/A";
        if ( t4.equals(""))
            t4 = "N/A";
        if ( t5.equals(""))
            t5 = "N/A";
        if ( t6.equals(""))
            t6 = "N/A";
        if ( t7.equals(""))
            t7 = "N/A";

```

```

        if ( t8.equals(""))
            t8 = "N/A";
        if ( t9.equals(""))
            t9 = "N/A";
        if ( t10.equals(""))
            t10 = "N/A";

        try {

            theJDBC.executeQuery("SELECT time FROM CycSet");
            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

            while (tokens.hasMoreTokens()){
                if (temp){
                    if
(! (tokens.nextToken().equals(choice1.getSelectedItem()))){
                        temp = true;
                    }
                    else
                        temp = false;
                } // end if
            } //END WHILE
        } //END TRY
        catch (SQLException sql) { ; }

        try {

            if (temp && (myWatch(choice1.getSelectedItem())) {

                theJDBC.executeUpdate("INSERT INTO CycSet VALUES

+ '(' + choice1.getSelectedItem() + ', '
+ "'" + t1 + "', "
+ "'" + t2 + "', "
+ "'" + t3 + "', "
+ "'" + t4 + "', "
+ "'" + t5 + "', "
+ "'" + t6 + "', "
+ "'" + t7 + "', "
+ "'" + t8 + "', "
+ "'" + t9 + "', "
+ "'" + t10 + "'" );

            } //end IF
            //else {
            //    showStatus("NOT ALLOWED! YOU TRIED TO MODIFY AN
EXISTING ENTRY OR NOT THE CURRENT PERIOD'S LOG");
            //} //END ELSE
        } //END TRY
        catch (SQLException sqlex) {; }

        textField67.setText("");
        textField68.setText("");
        textField69.setText("");
        textField70.setText("");
        textField71.setText("");
        textField72.setText("");

```

```

        textField73.setText("");
        textField74.setText("");
        textField75.setText("");
        textField76.setText("");

    } //end if

    /*******

if (source == button2) {

    String archArray [][] = null;

    StringTokenizer curTime = new
StringTokenizer(choice1.getSelectedItem(),"\n:",false);
    int t = Integer.parseInt(curTime.nextToken());

    try {

        theJDBC.executeQuery("SELECT time FROM CycSet");
        archArray = getArchive("CycSet",t);
    } //end try
    catch (SQLException sql) { ; }

    textField1.setText(archArray[5][0]+":00");
    textField2.setText(archArray[4][0]+":00");
    textField3.setText(archArray[3][0]+":00");
    textField4.setText(archArray[2][0]+":00");
    textField5.setText(archArray[1][0]+":00");
    textField6.setText(archArray[0][0]+":00");
    textField7.setText(archArray[5][1]);
    textField8.setText(archArray[5][2]);
    textField9.setText(archArray[5][3]);
    textField10.setText(archArray[5][4]);
    textField11.setText(archArray[5][5]);
    textField12.setText(archArray[5][6]);
    textField13.setText(archArray[5][7]);
    textField14.setText(archArray[5][8]);
    textField15.setText(archArray[5][9]);
    textField16.setText(archArray[5][10]);
    textField17.setText(archArray[4][1]);
    textField18.setText(archArray[4][2]);
    textField19.setText(archArray[4][3]);
    textField20.setText(archArray[4][4]);
    textField21.setText(archArray[4][5]);
    textField22.setText(archArray[4][6]);
    textField23.setText(archArray[4][7]);
    textField24.setText(archArray[4][8]);
    textField25.setText(archArray[4][9]);
    textField26.setText(archArray[4][10]);
    textField27.setText(archArray[3][1]);
    textField28.setText(archArray[3][2]);
    textField29.setText(archArray[3][3]);
    textField30.setText(archArray[3][4]);
    textField31.setText(archArray[3][5]);
    textField32.setText(archArray[3][6]);

```

```

textField33.setText(archArray[3][7]);
textField34.setText(archArray[3][8]);
textField35.setText(archArray[3][9]);
textField36.setText(archArray[3][10]);
textField37.setText(archArray[2][1]);
textField38.setText(archArray[2][2]);
textField39.setText(archArray[2][3]);
textField40.setText(archArray[2][4]);
textField41.setText(archArray[2][5]);
textField42.setText(archArray[2][6]);
textField43.setText(archArray[2][7]);
textField44.setText(archArray[2][8]);
textField45.setText(archArray[2][9]);
textField46.setText(archArray[2][10]);
textField47.setText(archArray[1][1]);
textField48.setText(archArray[1][2]);
textField49.setText(archArray[1][3]);
textField50.setText(archArray[1][4]);
textField51.setText(archArray[1][5]);
textField52.setText(archArray[1][6]);
textField53.setText(archArray[1][7]);
textField54.setText(archArray[1][8]);
textField55.setText(archArray[1][9]);
textField56.setText(archArray[1][10]);
textField57.setText(archArray[0][1]);
textField58.setText(archArray[0][2]);
textField59.setText(archArray[0][3]);
textField60.setText(archArray[0][4]);
textField61.setText(archArray[0][5]);
textField62.setText(archArray[0][6]);
textField63.setText(archArray[0][7]);
textField64.setText(archArray[0][8]);
textField65.setText(archArray[0][9]);
textField66.setText(archArray[0][10]);

} //end if

if (source == button4) {

    boolean temp = true;

    String t1 = textField155.getText();
    String t2 = textField156.getText();
    String t3 = textField157.getText();
    String t4 = textField158.getText();
    String t5 = textField159.getText();
    String t6 = textField160.getText();
    String t7 = textField161.getText();
    String t8 = textField162.getText();
    String t9 = textField163.getText();
    String t10 = textField164.getText();
    String t11 = textField165.getText();
    String t12 = textField165.getText();

    if ( t1.equals(""))
        t1 = "N/A";

```

```

        if ( t2.equals(""))
            t2 = "N/A";
        if ( t3.equals(""))
            t3 = "N/A";
        if ( t4.equals(""))
            t4 = "N/A";
        if ( t5.equals(""))
            t5 = "N/A";
        if ( t6.equals(""))
            t6 = "N/A";
        if ( t7.equals(""))
            t7 = "N/A";
        if ( t8.equals(""))
            t8 = "N/A";
        if ( t9.equals(""))
            t9 = "N/A";
        if ( t10.equals(""))
            t10 = "N/A";
        if ( t11.equals(""))
            t11 = "N/A";
        if ( t12.equals(""))
            t12 = "N/A";

        try {

            theJDBC.executeQuery("SELECT time FROM DetIn");
            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

            while (tokens.hasMoreTokens()){
                if (temp) {
                    if
(! (tokens.nextToken().equals(choice2.getSelectedItem())) )
                        temp = true;
                    else
                        temp = false;
                } //end if
            } //END WHILE

        } //END TRY
        catch (SQLException sql) { ; }

        try {
            if (temp ){ //&& (myWatch(choice2.getSelectedItem()))
{
                theJDBC.executeUpdate("INSERT INTO DetIn VALUES "
                    + "(" + choice2.getSelectedItem() + ", "
                    + "'" + t1 + "', "
                    + "'" + t2 + "', "
                    + "'" + t3 + "', "
                    + "'" + t4 + "', "
                    + "'" + t5 + "', "
                    + "'" + t6 + "', "
                    + "'" + t7 + "', "
                    + "'" + t8 + "', "
                    + "'" + t9 + "', "

```

```

        + "'" + t10 + "'", "
        + "'" + t11 + "'", "
        + "'" + t12 + "'" ) );
    } //end IF
} //END TRY
catch (SQLException sqlex) {; }

textField155.setText("");
textField156.setText("");
textField157.setText("");
textField158.setText("");
textField159.setText("");
textField160.setText("");
textField161.setText("");
textField162.setText("");
textField163.setText("");
textField164.setText("");
textField165.setText("");
textField166.setText("");

} //end if

//*****

if (source == button5) {

    String archArray [][] = null;

    StringTokenizer curTime = new
StringTokenizer(choice2.getSelectedItem(), "\n:", false);
    int t = Integer.parseInt(curTime.nextToken());

    try {
        theJDBC.executeQuery("SELECT time FROM DetIn");
        archArray = getArchive("DetIn", t);
    } //end try
    catch (SQLException sql) { ; }

    textField77.setText(archArray[5][0]+":00");
    textField78.setText(archArray[4][0]+":00");
    textField79.setText(archArray[3][0]+":00");
    textField80.setText(archArray[2][0]+":00");
    textField81.setText(archArray[1][0]+":00");
    textField82.setText(archArray[0][0]+":00");

    textField83.setText(archArray[5][1]);
    textField84.setText(archArray[5][2]);
    textField85.setText(archArray[5][3]);
    textField86.setText(archArray[5][4]);
    textField87.setText(archArray[5][5]);
    textField88.setText(archArray[5][6]);
    textField89.setText(archArray[5][7]);
    textField90.setText(archArray[5][8]);
    textField91.setText(archArray[5][9]);
    textField92.setText(archArray[5][10]);
    textField143.setText(archArray[5][11]);

```

```
textField149.setText(archArray[5][12]);
```

```
textField93.setText(archArray[4][1]);  
textField94.setText(archArray[4][2]);  
textField95.setText(archArray[4][3]);  
textField96.setText(archArray[4][4]);  
textField97.setText(archArray[4][5]);  
textField98.setText(archArray[4][6]);  
textField99.setText(archArray[4][7]);  
textField100.setText(archArray[4][8]);  
textField101.setText(archArray[4][9]);  
textField102.setText(archArray[4][10]);  
textField144.setText(archArray[4][11]);  
textField150.setText(archArray[4][12]);
```

```
textField103.setText(archArray[3][1]);  
textField104.setText(archArray[3][2]);  
textField105.setText(archArray[3][3]);  
textField106.setText(archArray[3][4]);  
textField107.setText(archArray[3][5]);  
textField108.setText(archArray[3][6]);  
textField109.setText(archArray[3][7]);  
textField110.setText(archArray[3][8]);  
textField111.setText(archArray[3][9]);  
textField112.setText(archArray[3][10]);  
textField145.setText(archArray[3][11]);  
textField151.setText(archArray[3][12]);
```

```
textField113.setText(archArray[2][1]);  
textField114.setText(archArray[2][2]);  
textField115.setText(archArray[2][3]);  
textField116.setText(archArray[2][4]);  
textField117.setText(archArray[2][5]);  
textField118.setText(archArray[2][6]);  
textField119.setText(archArray[2][7]);  
textField120.setText(archArray[2][8]);  
textField121.setText(archArray[2][9]);  
textField122.setText(archArray[2][10]);  
textField146.setText(archArray[2][11]);  
textField152.setText(archArray[2][12]);
```

```
textField123.setText(archArray[1][1]);  
textField124.setText(archArray[1][2]);  
textField125.setText(archArray[1][3]);  
textField126.setText(archArray[1][4]);  
textField127.setText(archArray[1][5]);  
textField128.setText(archArray[1][6]);  
textField129.setText(archArray[1][7]);  
textField130.setText(archArray[1][8]);  
textField131.setText(archArray[1][9]);  
textField132.setText(archArray[1][10]);  
textField147.setText(archArray[1][11]);  
textField153.setText(archArray[1][12]);
```

```
textField133.setText(archArray[0][1]);  
textField134.setText(archArray[0][2]);
```

```

        textField135.setText(archArray[0][3]);
        textField136.setText(archArray[0][4]);
        textField137.setText(archArray[0][5]);
        textField138.setText(archArray[0][6]);
        textField139.setText(archArray[0][7]);
        textField140.setText(archArray[0][8]);
        textField141.setText(archArray[0][9]);
        textField142.setText(archArray[0][10]);
        textField148.setText(archArray[0][11]);
        textField154.setText(archArray[0][12]);

    } //end if

//*****

    if (source == button6) {

        boolean temp = true;
        String curTabName = null;

        String t1 = textField210.getText();
        String t2 = textField211.getText();
        String t3 = textField212.getText();
        String t4 = textField213.getText();
        String t5 = textField214.getText();
        String t6 = textField215.getText();
        String t7 = textField209.getText();

        if ( t1.equals(""))
            t1 = "N/A";
        if ( t2.equals(""))
            t2 = "N/A";
        if ( t3.equals(""))
            t3 = "N/A";
        if ( t4.equals(""))
            t4 = "N/A";
        if ( t5.equals(""))
            t5 = "N/A";
        if ( t6.equals(""))
            t6 = "N/A";
        if ( t7.equals(""))
            t7 = "N/A";

        // I only used position 1 but anyone can add 7 more similar
        database table for
        // each position

        if ( choice5.getSelectedItem() == "1")
            curTabName = "AtmCon";
        else if ( choice5.getSelectedItem() == "2")
            curTabName = "AtmCon2";

        try {

            theJDBC.executeQuery("SELECT time FROM " + curTabName);

```

```

        StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

        while (tokens.hasMoreTokens()){
            if ( temp) {
                if
(! (tokens.nextToken().equals(choice3.getSelectedItem())) )
                    temp = true;
                else
                    temp = false;
            } //end if
        } //END WHILE
    } //END TRY
    catch (SQLException sql) { ; }

    try {
        if (temp) { //&& (myWatch(choice3.getSelectedItem()))
{
            theJDBC.executeUpdate("INSERT INTO " + curTabName + "
VALUES "
                + "(" + choice3.getSelectedItem() + ", "
                + "'" + t1 + "', "
                + "'" + t2 + "', "
                + "'" + t3 + "', "
                + "'" + t4 + "', "
                + "'" + t5 + "', "
                + "'" + t6 + "', "
                + "'" + t7 + "'" );

            } //end IF
        } //END TRY
        catch (SQLException sqllex) {; }

        textField210.setText("");
        textField211.setText("");
        textField212.setText("");
        textField213.setText("");
        textField214.setText("");
        textField215.setText("");
        textField209.setText("");

    } //end if

    /*******

    if (source == button7) {

        String archArray [][] = null;
        String curTabName = null;

        StringTokenizer curTime = new
StringTokenizer(choice3.getSelectedItem(), "\n:", false);
        int t = Integer.parseInt(curTime.nextToken());

```

```

        // I only used position 1 but anyone can add 7 more similar
database table for
        // each position
        if ( choice5.getSelectedItem() == "1")
            curTabName = "AtmCon";
        else if ( choice5.getSelectedItem() == "2")
            curTabName = "AtmCon2";

try {

    theJDBC.executeQuery("SELECT time FROM " + curTabName);
    archArray = getArchive(curTabName,t);
} //end try
catch (SQLException sql) { ; }

    textField167.setText(archArray[5][0]+":00");
    textField168.setText(archArray[4][0]+":00");
    textField169.setText(archArray[3][0]+":00");
    textField170.setText(archArray[2][0]+":00");
    textField171.setText(archArray[1][0]+":00");
    textField172.setText(archArray[0][0]+":00");

    textField173.setText(archArray[5][1]);
    textField179.setText(archArray[5][2]);
    textField185.setText(archArray[5][3]);
    textField191.setText(archArray[5][4]);
    textField197.setText(archArray[5][5]);
    textField203.setText(archArray[5][6]);

    textField174.setText(archArray[4][1]);
    textField180.setText(archArray[4][2]);
    textField186.setText(archArray[4][3]);
    textField192.setText(archArray[4][4]);
    textField198.setText(archArray[4][5]);
    textField204.setText(archArray[4][6]);

    textField175.setText(archArray[3][1]);
    textField181.setText(archArray[3][2]);
    textField187.setText(archArray[3][3]);
    textField193.setText(archArray[3][4]);
    textField199.setText(archArray[3][5]);
    textField205.setText(archArray[3][6]);

    textField176.setText(archArray[2][1]);
    textField182.setText(archArray[2][2]);
    textField188.setText(archArray[2][3]);
    textField194.setText(archArray[2][4]);
    textField200.setText(archArray[2][5]);
    textField206.setText(archArray[2][6]);

    textField177.setText(archArray[1][1]);
    textField183.setText(archArray[1][2]);
    textField189.setText(archArray[1][3]);
    textField195.setText(archArray[1][4]);
    textField201.setText(archArray[1][5]);

```

```

textField207.setText(archArray[1][6]);

textField178.setText(archArray[0][1]);
textField184.setText(archArray[0][2]);
textField190.setText(archArray[0][3]);
textField196.setText(archArray[0][4]);
textField202.setText(archArray[0][5]);
textField208.setText(archArray[0][6]);

} //end if

//*****

if (source == button3) {

    boolean temp = true;
    String curTabName = "remark";

    String t1 = textArea2.getText();

    if ( t1.equals(""))
        t1 = "N/A";

    try {

        theJDBC.executeQuery("SELECT time FROM " + curTabName);

        StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n", false);

        while (tokens.hasMoreTokens()){
            if (temp){
                if
(! (tokens.nextToken().equals(choice4.getSelectedItem()))){
                    temp = true;
                }
                else
                    temp = false;
            } //end if
        } //END WHILE
    } //END TRY
    catch (SQLException sql) { ; }

    try {
        if (temp ){ //&& (myWatch(choice4.getSelectedItem()))
{
            theJDBC.executeUpdate("INSERT INTO " + curTabName + "
VALUES "
                + "(" + choice4.getSelectedItem() + ", "
                + "'" + t1 + "'" );

        } //end IF
    } //END TRY
    catch (SQLException sqllex) {; }
}

```

```

        textArea2.setText("");

    } //end if

    /*******

    if (source == button8) {

        String archArray [][] = null;
        String curTabName = "remark";

        StringTokenizer curTime = new
StringTokenizer(choice4.getSelectedItem(), "\n:", false);
        int t = Integer.parseInt(curTime.nextToken());

        try {

            theJDBC.executeQuery("SELECT time FROM " + curTabName);
            archArray = getArchive(curTabName, t);
        } //end try
        catch (SQLException sql) { ; }

        textArea1.setText(archArray[0][0] + ":00 : " +
archArray[0][1]);

    } //end if
} // end action performed method

    /*******

    public String [][] getArchive (String tabName, int curTime){

        String t,k = null;
        String [][] realArray = null;
        int count = 0;

        try {

            theJDBC.executeQuery("SELECT * " + "FROM " + tabName);

            ResultSetMetaData md=theJDBC.theMetaData;
            ResultSet rs = theJDBC.theResultSet;
            int myColcount = md.getColumnCount();

            String temp [][] = new String[24][myColcount] ;
            realArray = new String [6][myColcount];

            StringTokenizer tokens = new StringTokenizer
(theJDBC.dumpResult(), "\n:", false);

            while (tokens.hasMoreTokens()){

                for (int i = 0 ; i < myColcount ; i++) {
                    if ( i == 0){
                        t=tokens.nextToken();
                        tokens.nextToken();

```

```

        //      if (t == null) .
        //          t = "0";
        temp[count][0] = t;

    } //end if
    else {
        k = tokens.nextToken();
        if (k == null)
            temp[count][i] = "N/A";
        else
            temp[count][i] = k;
    } // end else
    if ( Integer.parseInt(temp[count][0]) == curTime-1
)

        realArray[0][i] = temp[count][i];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-2)
        realArray[1][i] = temp[count][i];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-3)
        realArray[2][i] = temp[count][i];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-4)
        realArray[3][i] = temp[count][i];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-5)
        realArray[4][i] = temp[count][i];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-6)
        realArray[5][i] = temp[count][i];
    } // end for
    count++;
} //end while

if ((curTime-1) >= 0 )
    realArray[0][0] = Integer.toString(curTime-1);
else
    realArray[0][0] = Integer.toString(24+(curTime-1));

if ( (curTime-2) >= 0 )
    realArray[1][0] = Integer.toString(curTime-2);
else
    realArray[1][0] = Integer.toString(24+(curTime-2));

if ((curTime-3) >= 0 )
    realArray[2][0] = Integer.toString(curTime-3);
else
    realArray[2][0] = Integer.toString(24 +(curTime-3));

if ((curTime-4) >= 0 )
    realArray[3][0] = Integer.toString(curTime-4);
else
    realArray[3][0] = Integer.toString(24+(curTime-4));

if ((curTime-5) >= 0 )

```

```

        realArray[4][0] = Integer.toString(curTime-5);
    else
        realArray[4][0] = Integer.toString(24+(curTime-5));

    if ((curTime-6) >= 0 )
        realArray[5][0] = Integer.toString(curTime-6);
    else
        realArray[5][0] = Integer.toString(24+(curTime-6));

    for (int i = 0; i < 6;i++) {
        for (int j = 0; j < myColcount; j++) {
            if ( realArray[i][j] == null)

                realArray[i][j] = "N/A";
        } //end for
    } // end for
} //END TRY
catch (SQLException sql) { ; }

    return realArray;
} //end getArchive method

//*****

public boolean  myWatch (String s) {

    Date now;
    boolean temp = false;
    String message;

    now = new Date();
    message = now.getHours() + ":00";

    if (s.equals(message))
        temp = true;

    return temp;
} //end myWatch method

//*****
public void handleException(Exception e)
{
    e.printStackTrace( );
}

} // end class

```

APPENDIX E. LOG TAKING CLIENT SERVLET CODE

```
// Detectors Servlets
// New Attack Submarine Log Taking Servlet
// Version 1.2
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view Log actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/Log_Cover.htm
// Last Modified by (10\01\99)

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.lang.*;
import java.util.*;
import java.io.*;
import java.util.Date;

public class Log_detec extends HttpServlet
{
    String curTabName = "DetIn";
    String resultString [][]= null;
    boolean flag = true;

    public void doPost(HttpServletRequest req,
                       HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());

        // Print the HTML header
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Log Detector & Indicator Report</title>");
        out.println("</head>");
        out.println("<h2><center>");
        out.println("Log Detector & Indicator Report");
        out.println("<br>");
        out.println("(Press Back button to return)");
        out.println("</center></h2>");
        out.println("<br>");
    }
}
```

```

String values[];

// Get the location
String time = "";
values = req.getParameterValues("Selection");
if (values != null) {
    time = values[0];
}

// Get the position
String H1 = "";
values = req.getParameterValues("Field");
if (values != null) {
    H1 = values[0];
}
if ( H1.equals(""))
    H1="N/A";

// Get the position
String H2 = "";
values = req.getParameterValues("Field3");
if (values != null) {
    H2 = values[0];
}
if ( H2.equals(""))
    H2="N/A";

//Get man in charge
String WIC = "";
values = req.getParameterValues("Field4");
if (values != null) {
    WIC = values[0];
}
if ( WIC.equals(""))
    WIC="N/A";

//Get man in charge .
String AB = "";
values = req.getParameterValues("Field5");
if (values != null) {
    AB = values[0];
}
if ( AB.equals(""))
    AB="N/A";

//
String BC = "";
values = req.getParameterValues("Field6");
if (values != null) {
    BC = values[0];
}
if ( BC.equals(""))
    BC="N/A";

//

```

```

String AC = "";
values = req.getParameterValues("Field7");
if (values != null) {
    AC = values[0];
}
if ( AC.equals(""))
    AC="N/A";

//
String BatAgr = "";
values = req.getParameterValues("Field9");
if (values != null) {
    BatAgr = values[0];
}
if ( BatAgr.equals(""))
    BatAgr="N/A";

//
String BatFD = "";
values = req.getParameterValues("Field10");
if (values != null) {
    BatFD = values[0];
}
if ( BatFD.equals(""))
    BatFD="N/A";

//
String ICFuse = "";
values = req.getParameterValues("Field11");
if (values != null) {
    ICFuse = values[0];
}
if ( ICFuse.equals(""))
    ICFuse="N/A";

//
String SLmps = "";
values = req.getParameterValues("Field12");
if (values != null) {
    SLmps = values[0];
}
if ( SLmps.equals(""))
    SLmps="N/A";

//
String MK = "";
values = req.getParameterValues("Field13");
if (values != null) {
    MK = values[0];
}
if ( MK.equals(""))
    MK="N/A";

//
String EIC = "";
values = req.getParameterValues("Field14");
if (values != null) {

```

```

        EIC = values[0];
    }
    if ( EIC.equals(""))
        EIC="N/A";

    //
    String mychoice = "";
    values = req.getParameterValues("Selection16");
    if (values != null) {
        mychoice = values[0];
    }

    StringTokenizer curTime = new StringTokenizer(time, "\n:", false);
    int t = Integer.parseInt(curTime.nextToken());

    resultString = Archive(curTabName, t, true);

    if ( mychoice.equals("Archive")) {

        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println("<center><table border>");
        for (int i = 0; i < 7; i++) {
            out.println("<tr>");
            for (int ii = 0; ii < 13; ii++) {

                out.println("<td>"+resultString[i][ii]+"</td>");

            } // end for
            out.println("</tr>");
            //out.println("<br>");
        } // end for
        out.println("</table></center>");
        out.println("<br>");
        out.println("<br>");
        out.println("YOU ONLY RETREIVED PREVIOUS DOCUMENT.");
        out.println("IF YOU WANT TO SUBMIT DATA, PLEASE CHANGE THE
MODE");
    } //end if
    else{

        try {

            out.println("time = " + time + "<br>");
            out.println("H1 = " + H1 + "<br>");
            out.println("H2 = " + H2 + "<br>");
            out.println("WIC = " + WIC + "<br>");
            out.println("AB = " + AB + "<br>");
            out.println("BC = " + BC + "<br>");
            out.println("AC = " + AC + "<br>");

```

```

out.println("BatAgr = " + BatAgr + "<br>");
out.println("BatFD = " + BatFD + "<br>");
out.println("ICFuse = " + ICFuse + "<br>");
out.println("SLmps = " + SLmps + "<br>");
out.println("MK = " + MK + "<br>");
out.println("EIC = " + EIC + "<br>");
out.println("<br>");
out.println("<br>");

    if (flag){ // && (myWatch(time)) { //hour lock
        Connection con = null;
        Statement stmt = null;
        //flag = true;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
        con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
        stmt = con.createStatement();

        stmt.executeUpdate("INSERT INTO " + curTabName + " VALUES "
            + "(" + time + ", "
            + H1 + ", "
            + H2 + ", "
            + WIC + ", "
            + AB + ", "
            + BC + ", "
            + AC + ", "
            + BatAgr + ", "
            + BatFD + ", "
            + ICFuse + ", "
            + SLmps + ", "
            + MK + ", "
            + EIC + ")" );
        out.println("IF YOU WANT TO SEE PREVIOUS DATA,PLEASE
CHANGE THE MODE FIELD AT PREVIOUS PAGE");

        } //end if
        else {
            out.println("This hour's log is already entered. You can
not change the old entries.");
        } // end else

        if (!myWatch(time) && flag ) {
            out.println("<br>");
            out.println("You are trying to modify different period's
log. You are not allowed.");
        } // end if

    } // end try
    catch (Exception ex) {
        out.println("Exception!");
        //ex.printStackTrace(out);
        out.println("Please check your data. All boxes have to be
filled up.");
    }

```

```

    } // end catch
    } // end if
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
} // end post method

//*****

    public String [][] Archive ( String tabName, int curTime, boolean
resp) throws ServletException, java.io.IOException{

    String t,k = null;
    String [][] realArray = null;
    int count = 0;

    flag = resp;

    try {

        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

        con = DriverManager.getConnection("jdbc:odbc:db1");

        stmt = con.createStatement();

        rs = stmt.executeQuery("SELECT * " + "FROM " + tabName);

        ResultSetMetaData md=rs.getMetaData();
        int myColcount = md.getColumnCount();

        String temp [][] = new String[24][myColcount] ;
        realArray = new String [7][myColcount];

        while (rs.next()){
            for (int i = 1 ; i <= myColcount ; i++) {

                k = rs.getString(i);

                if ( i == 1){

```

```

StringTokenizer arcTime = new StringTokenizer(
k, "\n:", false);

int l = Integer.parseInt(arcTime.nextToken());
// to prevent multiple entry regulates the
flag

    if(flag){
        if (!(l == curTime)){
            flag = true;
            //break;
        } // end if
        else
            flag = false;
    } // end if

    temp[count][0] = Integer.toString(l);

} //end if
else {

    if (k == null)
        temp[count][i-1] = "N/A";
    else
        temp[count][i-1] = k;
} // end else

if ( Integer.parseInt(temp[count][0]) == curTime-1
)

    realArray[1][i-1] = temp[count][i-1];
else if ( Integer.parseInt(temp[count][0]) ==
curTime-2)

    realArray[2][i-1] = temp[count][i-1];
else if ( Integer.parseInt(temp[count][0]) ==
curTime-3)

    realArray[3][i-1] = temp[count][i-1];
else if ( Integer.parseInt(temp[count][0]) ==
curTime-4)

    realArray[4][i-1] = temp[count][i-1];
else if ( Integer.parseInt(temp[count][0]) ==
curTime-5)

    realArray[5][i-1] = temp[count][i-1];
else if ( Integer.parseInt(temp[count][0]) ==
curTime-6)

    realArray[6][i-1] = temp[count][i-1];

} // end for

count++;
} //end while

//realArray[0][0] = md.getColumnLabel(1);
if ( (curTime-1) >= 0 )
    realArray[1][0] = Integer.toString(curTime-1);
else
    realArray[1][0] = Integer.toString(24+(curTime-1));

```

```

        if ( (curTime-2) >= 0 )
            realArray[2][0] = Integer.toString(curTime-2);
        else
            realArray[2][0] = Integer.toString(24+(curTime-2));

        if ((curTime-3) >= 0 )
            realArray[3][0] = Integer.toString(curTime-3);
        else
            realArray[3][0] = Integer.toString(24 +(curTime-3));

        if ((curTime-4) >= 0 )
            realArray[4][0] = Integer.toString(curTime-4);
        else
            realArray[4][0] = Integer.toString(24+(curTime-4));

        if ((curTime-5) >= 0 )
            realArray[5][0] = Integer.toString(curTime-5);
        else
            realArray[5][0] = Integer.toString(24+(curTime-5));

        if ((curTime-6) >= 0 )
            realArray[6][0] = Integer.toString(curTime-6);
        else
            realArray[6][0] = Integer.toString(24+(curTime-6));

        for (int i = 0; i < 7;i++) {
            for (int j = 0; j <= myColcount-1; j++) {
                if ( i == 0)
                    realArray[0][j] = md.getColumnLabel(j+1);
                if ( realArray[i][j].equals( null))
                    realArray[i][j] = "N/A";
            } //end for
        } // end for

    } //END TRY
    catch (Exception ex) {

    }
    //catch (SQLException sql) { ; }

    return realArray;
} //end getArchive method

//*****

public boolean myWatch (String s) {

    Date now;

```

```

        boolean temp = false;
        String message;

        now = new Date();
        message = now.getHours() + ":00";

        if (s.equals(message))
            temp = true;

        return temp;
    } //end myWatch method

} // end servlet

*****

*****

// Log_cycleSet Servlet
// New Attack Submarine Log Taking Servlet
// Version 1.2
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view Log actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/Log_Cover.htm
// Last Modified by (10\01\99)

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.lang.*;
import java.util.*;
import java.io.*;
import java.util.Date;

public class Log_cycleSet extends HttpServlet
{
    String curTabName = "CycSet";
    String resultString [][]= null;
    boolean flag = true;

    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

```

```

// Create a PrintWriter to write the response
java.io.PrintWriter out =
    new java.io.PrintWriter(resp.getOutputStream());

// Print the HTML header
out.println("<html>");
out.println("<head>");
out.println("<title>Log 400 Cycle Set Report</title>");
out.println("</head>");
out.println("<h2><center>");
out.println("Log 400 Cycle Set Report");
out.println("(Press Back button to return)");
out.println("</center></h2>");
out.println("<br>");

String values[];

// Get the location
String time = "";
values = req.getParameterValues("Selection");
if (values != null) {
    time = values[0];
}

// Get the position
String dc1 = "";
values = req.getParameterValues("Field");
if (values != null) {
    dc1 = values[0];
}
if (dc1.equals(""))
    dc1="N/A";

// Get the position
String dc2 = "";
values = req.getParameterValues("Field3");
if (values != null) {
    dc2 = values[0];
}
if (dc2.equals(""))
    dc2="N/A";

//Get man in charge
String ac1 = "";
values = req.getParameterValues("Field4");
if (values != null) {
    ac1 = values[0];
}
if (ac1.equals(""))
    ac1 = "N/A";

//Get man in charge
String ac2 = "";
values = req.getParameterValues("Field8");
if (values != null) {

```

```

        ac2 = values[0];
    }
    if ( ac2.equals(""))
        ac2 = "N/A";

    //
    String freq1 = "";
    values = req.getParameterValues("Field5");
    if (values != null) {
        freq1 = values[0];
    }
    if ( freq1.equals(""))
        freq1 = "N/A";

    //
    String freq2 = "";
    values = req.getParameterValues("Field9");
    if (values != null) {
        freq2 = values[0];
    }
    if ( freq2.equals(""))
        freq2 = "N/A";

    //
    String volt1 = "";
    values = req.getParameterValues("Field6");
    if (values != null) {
        volt1 = values[0];
    }
    if ( volt1.equals(""))
        volt1 = "N/A";

    //
    String volt2 = "";
    values = req.getParameterValues("Field10");
    if (values != null) {
        volt2 = values[0];
    }
    if ( volt2.equals(""))
        volt2 = "N/A";

    //
    String grnd1 = "";
    values = req.getParameterValues("Field7");
    if (values != null) {
        grnd1 = values[0];
    }
    if ( grnd1.equals(""))
        grnd1 = "N/A";

    //
    String grnd2 = "";
    values = req.getParameterValues("Field11");
    if (values != null) {
        grnd2 = values[0];
    }

```

```

if ( grnd2.equals(""))
    grnd2 = "N/A";

//
String mychoice = "";
values = req.getParameterValues("Selection14");
if (values != null) {
    mychoice = values[0];
}

StringTokenizer curTime = new StringTokenizer(time, "\n:", false);
int t = Integer.parseInt(curTime.nextToken());

resultString = Archive(curTabName, t, true);

if ( mychoice.equals("Archive")) {

    out.println("<br>");
    out.println("<br>");
    out.println("<br>");
    out.println("<center><table border>");
    for (int i = 0; i < 7; i++) {
        out.println("<tr>");
        for (int ii = 0; ii < 11; ii++) {

            out.println("<td>"+resultString[i][ii]+"</td>");

        } // end for
        out.println("</tr>");
        //out.println("<br>");
    } // end for
    out.println("</table></center>");
    out.println("<br>");
    out.println("<br>");
    out.println("YOU ONLY RETREIVED PREVIOUS DOCUMENT.");
    out.println("IF YOU WANT TO SUBMIT DATA, PLEASE CHANGE THE
MODE");
} //end if
else{

    try {

        out.println("time = " + time + "<br>");
        out.println("dc1 = " + dc1 + "<br>");
        out.println("ac1 = " + ac1 + "<br>");
        out.println("freq1 = " + freq1 + "<br>");
        out.println("volt1 = " + volt1 + "<br>");
        out.println("grnd1 = " + grnd1 + "<br>");
        out.println("dc2 = " + dc2 + "<br>");
        out.println("ac2 = " + ac2 + "<br>");
    }
}

```

```

out.println("freq2 = " + freq2 + "<br>");
out.println("volt2 = " + volt2 + "<br>");
out.println("grnd2 = " + grnd2 + "<br>");
out.println("<br>");
out.println("<br>");

    if (flag && (myWatch(time))) { //hour lock
        Connection con = null;
        Statement stmt = null;
        //flag = true;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
        con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
        stmt = con.createStatement();

        stmt.executeUpdate("INSERT INTO " + curTabName + " VALUES "
            + "(" + time + "', "
            + "'" + dc1 + "', "
            + "'" + ac1 + "', "
            + "'" + freq1 + "', "
            + "'" + volt1 + "', "
            + "'" + grnd1 + "', "
            + "'" + dc2 + "', "
            + "'" + ac2 + "', "
            + "'" + freq2 + "', "
            + "'" + volt2 + "', "
            + "'" + grnd2 + "'" );
        out.println("IF YOU WANT TO SEE PREVIOUS DATA, PLEASE
CHANGE THE MODE FIELD AT PREVIOUS PAGE");

        } //end if
        else {
            out.println("This hour's log is already entered. You can
not change the old entries.");
        } // end else

        if (!myWatch(time) && flag ) {
            out.println("<br>");
            out.println("You are trying to modify different period's
log. You are not allowed.");
        } // end if

        } // end try
        catch (Exception ex) {
            out.println("Exception!");
            //ex.printStackTrace(out);
            out.println("Please check your data. All boxes have to be
filled up.");
        } // end catch
    } //end if
} //end if
// Wrap up
out.println("</html>");
out.flush();

```

```

        out.close();
    } // end post method

//*****

    public String [][] Archive ( String tabName, int curTime,boolean
resp) throws ServletException, java.io.IOException{

        String t,k = null;
        String [][] realArray = null;
        int count = 0;

        flag = resp;

        try {

            Connection con = null;
            Statement stmt = null;
            ResultSet rs = null;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

            con = DriverManager.getConnection("jdbc:odbc:db1");

            stmt = con.createStatement();

            rs = stmt.executeQuery("SELECT * " + "FROM " +  tabName);

            ResultSetMetaData md=rs.getMetaData();
            int myColcount = md.getColumnCount();

            String temp [][] = new String[24][myColcount] ;
            realArray = new String  [7][myColcount];

            while (rs.next()){
                for (int i = 1 ; i <= myColcount ; i++) {

                    k = rs.getString(i);

                    if ( i == 1){
                        StringTokenizer arcTime = new StringTokenizer(
k, "\n:", false);
                        int l = Integer.parseInt(arcTime.nextToken());
                        // to prevent multiple entry regulates the
flag
                        if(flag){

```

```

        if (!(l == curTime)){
            flag = true;
            //break;
        } // end if
        else
            flag = false;
    } // end if

    temp[count][0] = Integer.toString(l);

} //end if
else {

    if (k == null)
        temp[count][i-1] = "N/A";
    else
        temp[count][i-1] = k;
} // end else

    if ( Integer.parseInt(temp[count][0]) == curTime-1
)
        realArray[1][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-2)
        realArray[2][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-3)
        realArray[3][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-4)
        realArray[4][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-5)
        realArray[5][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-6)
        realArray[6][i-1] = temp[count][i-1];

} // end for

    count++;
} //end while

//realArray[0][0] = md.getColumnLabel(1);
if ((curTime-1) >= 0 )
    realArray[1][0] = Integer.toString(curTime-1);
else
    realArray[1][0] = Integer.toString(24+(curTime-1));

if ( (curTime-2) >= 0 )
    realArray[2][0] = Integer.toString(curTime-2);
else
    realArray[2][0] = Integer.toString(24+(curTime-2));

```

```

        if ((curTime-3) >= 0 )
            realArray[3][0] = Integer.toString(curTime-3);
        else
            realArray[3][0] = Integer.toString(24 +(curTime-3));

        if ((curTime-4) >= 0 )
            realArray[4][0] = Integer.toString(curTime-4);
        else
            realArray[4][0] = Integer.toString(24+(curTime-4));

        if ((curTime-5) >= 0 )
            realArray[5][0] = Integer.toString(curTime-5);
        else
            realArray[5][0] = Integer.toString(24+(curTime-5));

        if ((curTime-6) >= 0 )
            realArray[6][0] = Integer.toString(curTime-6);
        else
            realArray[6][0] = Integer.toString(24+(curTime-6));

        for (int i = 0; i < 6;i++) {
            for (int j = 0; j <= myColcount-1; j++) {
                if ( i == 0)
                    realArray[0][j] = md.getColumnLabel(j+1);
                if ( realArray[i][j].equals( null))
                    realArray[i][j] = "N/A";
            } //end for
        } // end for

    } //END TRY
    catch (Exception ex) {

    }
    //catch (SQLException sql) { ; }

    return realArray;
} //end getArchive method

//*****

public boolean  myWatch (String s) {

    Date now;
    boolean temp = false;
    String message;

    now = new Date();
    message = now.getHours() + ":00";

```

```

        if (s.equals(message))
            temp = true;

        return temp;
    } //end myWatch method

} // end servlet

*****

*****

// Log Atmosphere Servlet
// New Attack Submarine Log Taking Servlet
// Version 1.1
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view Log actions
// for specified US Naval Vessels (Simulated)
// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/Log_Cover.htm
// Last Modified by (09\02\99)

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.lang.*;
import java.util.*;
import java.io.*;
import java.util.Date;

public class Log_ClientServlet extends HttpServlet
{
    String curTabName = null;
    String resultString [][]= null;
    boolean flag = true;

    public void doPost(HttpServletRequest req,
                        HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());

        // Print the HTML header

```

```

out.println("<html>");
out.println("<head>");
out.println("<title>Log Report</title>");
out.println("</head>");
out.println("<h2><center>");
out.println("Atm Control Log");
out.println("<br>");
out.println("(Press Back button to return)");
out.println("</center></h2>");
out.println("<br>");

String values[];

// Get the position
String pos = "";
values = req.getParameterValues("Selection");
if (values != null) {
    pos = values[0];
}

// Get the location
String time = "";
values = req.getParameterValues("Selection2");
if (values != null) {
    time = values[0];
}

// get state of fire
String CO = "";
values = req.getParameterValues("Field");
if (values != null) {
    CO = values[0];
}
if ( CO.equals(""))
    CO = "N/A";

//Get man in charge
String R_114 = "";
values = req.getParameterValues("Field4");
if (values != null) {
    R_114 = values[0];
}
if ( R_114.equals(""))
    R_114 = "N/A";

//
String R_12 = "";
values = req.getParameterValues("Field5");
if (values != null) {
    R_12 = values[0];
}
if ( R_12.equals(""))
    R_12 = "N/A";

//

```

```

String O2 = "";
values = req.getParameterValues("Field6");
if (values != null) {
    O2 = values[0];
}
if ( O2.equals(""))
    O2 = "N/A";

//
String CO2 = "";
values = req.getParameterValues("Field7");
if (values != null) {
    CO2 = values[0];
}
if ( CO2.equals(""))
    CO2 = "N/A";

//
String H2 = "";
values = req.getParameterValues("Field8");
if (values != null) {
    H2 = values[0];
}
if ( H2.equals(""))
    H2 = "N/A";

//
String TotPres = "";
values = req.getParameterValues("Field9");
if (values != null) {
    TotPres = values[0];
}
if ( TotPres.equals(""))
    TotPres = "N/A";

//
String mychoice = "";
values = req.getParameterValues("Selection12");
if (values != null) {
    mychoice = values[0];
}

// I only used position 1 but anyone can add 7 more similar
database table for
// each position

if ( pos.equals("1"))
    curTabName = "AtmCon";
else if (pos.equals("2"))
    curTabName = "AtmCon2";

StringTokenizer curTime = new StringTokenizer(time, "\n:", false);
int t = Integer.parseInt(curTime.nextToken());

resultString = Archive(curTabName, t, true);

```

```

if ( mychoice.equals("Archive")) {

    out.println("<br>");
    out.println("<br>");
    out.println("<br>");
    out.println("<center><table border>");
    for (int i = 0; i < 7; i++) {
        out.println("<tr>");
        for (int ii = 0; ii < 8; ii++) {

            out.println("<td>"+resultString[i][ii]+"</td>");

        } // end for
        out.println("</tr>");
        //out.println("<br>");
    } // end for
    out.println("</table></center>");
    out.println("<br>");
    out.println("<br>");
    out.println("YOU ONLY RETREIVED PREVIOUS DOCUMENT.");
    out.println("IF YOU WANT TO SUBMIT DATA, PLEASE CHANGE THE
MODE");
} //end if
else{

    try {

        out.println("Position = " + pos + "<br>");
        out.println("time = " + time + "<br>");
        out.println("CO = " + CO + "<br>");
        out.println("R-114 = " + R_114 + "<br>");
        out.println("R-12 = " + R_12 + "<br>");
        out.println("O2 = " + O2 + "<br>");
        out.println("CO2 = " + CO2 + "<br>");
        out.println("H2 = " + H2 + "<br>");
        out.println("Total Press = " + TotPres + "<br>");
        out.println("<br>");
        out.println("<br>");

        if (flag){ // && (myWatch(time)) { //hour lock
            Connection con = null;
            Statement stmt = null;
            //flag = true;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
            con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
            stmt = con.createStatement();

            stmt.executeUpdate("INSERT INTO " + curTabName + " VALUES "
                + "(" + time + ", "

```

```

        + "'" + CO + "'", "
        + "'" + R_114 + "'", "
        + "'" + R_12 + "'", "
        + "'" + O2 + "'", "
        + "'" + CO2 + "'", "
        + "'" + H2 + "'", "
        + "'" + TotPres + "'" ) );
    out.println("IF YOU WANT TO SEE PREVIOUS DATA, PLEASE
CHANGE THE MODE FIELD AT PREVIOUS PAGE");

    } //end if
    else {
        out.println("This hour's log is already entered. You can
not change the old entries.");
    } // end else

    if (!myWatch(time) && flag ) {
        out.println("<br>");
        out.println("You are trying to modify different period's
log. You are not allowed.");
    } // end if

    } // end try
    catch (Exception ex) {
        out.println("Exception!");
        //ex.printStackTrace(out);
        out.println("Please check your data. All boxes have to be
filled up.");
    } // end catch
    } //end if
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
} // end post method

//*****

    public String [][] Archive ( String tabName, int curTime,boolean
resp) throws ServletException, java.io.IOException{

    String t,k = null;
    String [][] realArray = null;
    int count = 0;

    flag = resp;

    try {

        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

con = DriverManager.getConnection("jdbc:odbc:db1");

stmt = con.createStatement();

rs = stmt.executeQuery("SELECT * " + "FROM " + tabName);


ResultSetMetaData md=rs.getMetaData();
int myColcount = md.getColumnCount();

String temp [][] = new String[24][myColcount] ;
realArray = new String [7][myColcount];


while (rs.next()){
    for (int i = 1 ; i <= myColcount ; i++) {

        k = rs.getString(i);

        if ( i == 1){
            StringTokenizer arcTime = new StringTokenizer(
k, "\n:", false);

            int l = Integer.parseInt(arcTime.nextToken());
            // to prevent multiple entry regulates the

            flag

            if(flag){
                if (!(l == curTime)){
                    flag = true;
                    //break;
                } // end if
            } else
                flag = false;
            } // end if

            temp[count][0] = Integer.toString(l);

        } //end if
        else {

            if (k == null)
                temp[count][i-1] = "N/A";
            else
                temp[count][i-1] = k;
        } // end else

        if ( Integer.parseInt(temp[count][0]) == curTime-1
    )

```

```

        realArray[1][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-2)
        realArray[2][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-3)
        realArray[3][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-4)
        realArray[4][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-5)
        realArray[5][i-1] = temp[count][i-1];
    else if ( Integer.parseInt(temp[count][0]) ==
curTime-6)
        realArray[6][i-1] = temp[count][i-1];

    } // end for

    count++;
} //end while

//realArray[0][0] = md.getColumnLabel(1);
if ( (curTime-1) >= 0 )
    realArray[1][0] = Integer.toString(curTime-1);
else
    realArray[1][0] = Integer.toString(24+(curTime-1));

if ( (curTime-2) >= 0 )
    realArray[2][0] = Integer.toString(curTime-2);
else
    realArray[2][0] = Integer.toString(24+(curTime-2));

if ((curTime-3) >= 0 )
    realArray[3][0] = Integer.toString(curTime-3);
else
    realArray[3][0] = Integer.toString(24 +(curTime-3));

if ((curTime-4) >= 0 )
    realArray[4][0] = Integer.toString(curTime-4);
else
    realArray[4][0] = Integer.toString(24+(curTime-4));

if ((curTime-5) >= 0 )
    realArray[5][0] = Integer.toString(curTime-5);
else
    realArray[5][0] = Integer.toString(24+(curTime-5));

if ((curTime-6) >= 0 )
    realArray[6][0] = Integer.toString(curTime-6);
else
    realArray[6][0] = Integer.toString(24+(curTime-6));

for (int i = 0; i < 7;i++) {
    for (int j = 0; j <= myColcount-1; j++) {
        if ( i == 0)

```

```

        realArray[0][j] = md.getColumnLabel(j+1);
        if ( realArray[i][j].equals( null))
            realArray[i][j] = "N/A";
    } //end for
} // end for

} //END TRY
catch (Exception ex) {

}
//catch (SQLException sql) { ; }

return realArray;
} //end getArchive method

//*****

public boolean myWatch (String s) {

    Date now;
    boolean temp = false;
    String message;

    now = new Date();
    message = now.getHours() + ":00";

    if (s.equals(message))
        temp = true;

    return temp;
} //end myWatch method
} // end servlet

*****

*****

// Log Remarks Servlet
// New Attack Submarine Log Taking Servlet
// Version 1.2
// 1st Lieutenant Hanceri SAYAT
// Servlet allows the user to update and view Log actions
// for specified US Naval Vessels (Simulated)

```

```

// Contains no classified or sensitive data
// Servlet can be viewed at:
// http://131.120.27.67:8080/Log_Cover.htm
// Last Modified by (10\01\99)

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.lang.*;
import java.util.*;
import java.io.*;
import java.util.Date;

public class Log_remark extends HttpServlet
{
    String curTabName = "remark";
    String resultString [][]= null;
    boolean flag = true;

    public void doPost(HttpServletRequest req,
                       HttpServletResponse resp)
        throws ServletException, java.io.IOException
    {
        // Set the content type of the response
        resp.setContentType("text/html");

        // Create a PrintWriter to write the response
        java.io.PrintWriter out =
            new java.io.PrintWriter(resp.getOutputStream());

        // Print the HTML header
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Log Remarks Report</title>");
        out.println("</head>");
        out.println("<h2><center>");
        out.println("Remarks Report");
        out.println("<br>");
        out.println("Press Back button to return");
        out.println("</center></h2>");
        out.println("<br>");

        String values[];

        // Get the location
        String time = "";
        values = req.getParameterValues("Selection");
        if (values != null) {
            time = values[0];
        }

        // Get the position
        String remark = "";
    }
}

```

```

values = req.getParameterValues("TextArea");
if (values != null) {
    remark = values[0];
}
if ( remark.equals(""))
    remark="N/A";

//
String mychoice = "";
values = req.getParameterValues("Selection5");
if (values != null) {
    mychoice = values[0];
}

StringTokenizer curTime = new StringTokenizer(time, "\n:", false);
int t = Integer.parseInt(curTime.nextToken());

resultString = Archive(curTabName, t, true);

if ( mychoice.equals("Archive")) {

    out.println("<br>");
    out.println("<br>");
    out.println("<br>");
    out.println("<center><table border>");
    for (int i = 0; i < 7; i++) {
        out.println("<tr>");
        for (int ii = 0; ii < 2; ii++) {

            out.println("<td>"+resultString[i][ii]+"</td>");

        } // end for
        out.println("</tr>");
        //out.println("<br>");
    } // end for
    out.println("</table></center>");
    out.println("<br>");
    out.println("<br>");
    out.println("YOU ONLY RETREIVED PREVIOUS DOCUMENT.");
    out.println("IF YOU WANT TO SUBMIT DATA, PLEASE CHANGE THE
MODE");
} //end if
else{

    try {

        out.println("time = " + time + "<br>");
        out.println("Remark = " + remark + "<br>");
        out.println("<br>");
        out.println("<br>");
    }
}

```

```

        if (flag){ // && (myWatch(time)) { //hour lock
            Connection con = null;
            Statement stmt = null;
            //flag = true;

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
            con =
DriverManager.getConnection("jdbc:odbc:db1",null,null);
            stmt = con.createStatement();

            stmt.executeUpdate("INSERT INTO " + curTabName + " VALUES "
                + "(" + time + ", "
                + "'" + remark + "'" + ")" );
            out.println("IF YOU WANT TO SEE PREVIOUS DATA, PLEASE
CHANGE THE MODE FIELD AT PREVIOUS PAGE");

        } //end if
        else {
            out.println("This hour's log is already entered. You can
not change the old entries.");
        } // end else

        if (!myWatch(time) && flag ) {
            out.println("<br>");
            out.println("You are trying to modify different period's
log. You are not allowed.");
        } // end if

        } // end try
        catch (Exception ex) {
            out.println("Exception!");
            //ex.printStackTrace(out);
            out.println("Please check your data. All boxes have to be
filled up.");
        } // end catch
    } //end if
    // Wrap up
    out.println("</html>");
    out.flush();
    out.close();
} // end post method

//*****

    public String [][] Archive ( String tabName, int curTime,boolean
resp) throws ServletException, java.io.IOException{

        String t,k = null;
        String [][] realArray = null;
        int count = 0;

        flag = resp;

```

```

try {

    Connection con = null;
    Statement stmt = null;
    ResultSet rs = null;

    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

    con = DriverManager.getConnection("jdbc:odbc:db1");

    stmt = con.createStatement();

    rs = stmt.executeQuery("SELECT * " + "FROM " + tabName);

    ResultSetMetaData md=rs.getMetaData();
    int myColcount = md.getColumnCount();

    String temp [][] = new String[24][myColcount] ;
    realArray = new String  [7][myColcount];

    while (rs.next()){
        for (int i = 1 ; i <= myColcount ; i++) {

            k = rs.getString(i);

            if ( i == 1){
                StringTokenizer arcTime = new StringTokenizer(
k, "\n:", false);

                int l = Integer.parseInt(arcTime.nextToken());
                // to prevent multiple entry regulates the
flag

                if(flag){
                    if (!(l == curTime)){
                        flag = true;
                        //break;
                    } // end if
                } else
                    flag = false;
            } // end if

            temp[count][0] = Integer.toString(l);

        } //end if
        else {

            if (k == null)
                temp[count][i-1] = "N/A";

```

```

        else
            temp[count][i-1] = k;
        } // end else

        if ( Integer.parseInt(temp[count][0]) == curTime-1
    )
        realArray[1][i-1] = temp[count][i-1];
    curTime-2) else if ( Integer.parseInt(temp[count][0]) ==
        realArray[2][i-1] = temp[count][i-1];
    curTime-3) else if ( Integer.parseInt(temp[count][0]) ==
        realArray[3][i-1] = temp[count][i-1];
    curTime-4) else if ( Integer.parseInt(temp[count][0]) ==
        realArray[4][i-1] = temp[count][i-1];
    curTime-5) else if ( Integer.parseInt(temp[count][0]) ==
        realArray[5][i-1] = temp[count][i-1];
    curTime-6) else if ( Integer.parseInt(temp[count][0]) ==
        realArray[6][i-1] = temp[count][i-1];

        } // end for

        count++;
    } //end while

    //realArray[0][0] = md.getColumnLabel(1);
    if ( (curTime-1) >= 0 )
        realArray[1][0] = Integer.toString(curTime-1);
    else
        realArray[1][0] = Integer.toString(24+(curTime-1));

    if ( (curTime-2) >= 0 )
        realArray[2][0] = Integer.toString(curTime-2);
    else
        realArray[2][0] = Integer.toString(24+(curTime-2));

    if ((curTime-3) >= 0 )
        realArray[3][0] = Integer.toString(curTime-3);
    else
        realArray[3][0] = Integer.toString(24 +(curTime-3));

    if ((curTime-4) >= 0 )
        realArray[4][0] = Integer.toString(curTime-4);
    else
        realArray[4][0] = Integer.toString(24+(curTime-4));

    if ((curTime-5) >= 0 )
        realArray[5][0] = Integer.toString(curTime-5);
    else
        realArray[5][0] = Integer.toString(24+(curTime-5));

```

```

        if ((curTime-6) >= 0 )
            realArray[6][0] = Integer.toString(curTime-6);
        else
            realArray[6][0] = Integer.toString(24+(curTime-6));

        for (int i = 0; i < 7;i++) {
            for (int j = 0; j <= myColcount-1; j++) {
                if ( i == 0)
                    realArray[0][j] = md.getColumnLabel(j+1);
                if ( realArray[i][j].equals( null))
                    realArray[i][j] = "N/A";
            } //end for
        } // end for

    } //END TRY
    catch (Exception ex) {

    }
    //catch (SQLException sql) { ; }

    return realArray;
} //end getArchive method

//*****

public boolean myWatch (String s) {

    Date now;
    boolean temp = false;
    String message;

    now = new Date();
    message = now.getHours() + ":00";

    if (s.equals(message))
        temp = true;

    return temp;
} //end myWatch method

//*****

} // end servlet

```

LIST OF REFERENCES

1. Yun, Xiaoping & Lewis, Ted, *Feasibility Analysis of Deploying Wireless LAN Onboard Submarines and Surface Ships*, Naval Postgraduate School Research Magazine, Volume 9, pg.1, Monterey, CA, 1999.
2. Dayem, Rifaat A., *Mobile Data and Wireless LAN Technologies*, Prentice Hall, Upper Saddle River, NJ, 1997.
3. Webb, William, *Introduction to Wireless Local Loop*, Artech House Inc., Cambridge, MA, 1998.
4. Stallings, William, *Data and Computer Communications*, Prentice Hall, Upper Saddle River, NJ, 1997.
5. Rappaport, T., *Wireless Communication Principles and Practice*, Prentice Hall, Upper Saddle River, NJ, 1996.
6. Black, Uyles, *Mobile and Wireless Networks*, Prentice Hall, Upper Saddle River, NJ, 1996.
7. Cummiskey, James, *Internetworking*, Master's Thesis, Naval Postgraduate School, Monterey, CA, 1996.
8. Wu, Thomas C., *An Introduction to Objected-Oriented Programming with Java*, McGraw Hill, New York, NY, 1999.
9. Reese, George, *Database Programming with JDBC and Java*, O'Reilly, Sebastopol, CA, 1997.
10. Hamilton, Graham & Cattell, Rick, *JDBC Database Access with Java*, Addison-Wesley, Mountain View, CA, 1997.
11. Garingo, Gary D., *Java Based Data Connectivity*, Master's Thesis, Naval Postgraduate School, Monterey CA, 1997.
12. Hamilton, G. & Cattell, R. & Fisher, Maydene, *JDBC Database Access with Java*, Addison-Wesley Longman Inc., Reading, MA, 1997.
13. Culwin, Fintan, *A Java GUI Programmer's Primer*, Prentice Hall, Upper Saddle River, NJ, 1998.

14. Rothenhaus, Kurt, *Software Applications in Java for Pen Based Portable Processors Operating On a Wireless LAN*, Master's Thesis, Naval Postgraduate School, Monterey, CA, 1999.
15. Holzschlag, Molly E., *Using HTML 4*, QUE Corporation, Indianapolis, IN, 1999.
16. Hunter, Jason, *Java Servlet Programming*, O'Reilly Assoc., Sebastopol, CA, 1998.
17. Hitachi, HPW-600ET Series Handheld PC Manual, San Diego, CA, 1999.
18. Lucent Technologis, WaveLAN IEEE 802.11 PC Card Manual, Nieuwegein, The Netherlands, 1999.
19. Proxim, RangeLAN2 7400 Wireless LAN Adapter Manual, Mountain View, CA, 1997.
20. Mitsubishi, Amity VP Hanheld Computer Manual, Japan, 1997.
21. Casio, Cassiopedia PA-2400 Manual, Tokyo, Japan, 1998.
22. Chan, Patrick & Lee, Rosanna, *Java Class Libraries I - II*, Addison-Wesley Longman Inc., Reading, MA, 1998.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center..... 2
8725 John J. Kingman Road, Suite 0944
Fort Belvoir, VA 22060-6218

2. Dudley Knox Library 2
Naval Postgraduate School
411 Dyer Rd.
Monterey, CA 93943

3. Chairman, Code EC.....1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121

4. Xiaoping Yun, Code EE/X. 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121

5. C. Thomas Wu, Code CS/Wq 1
Department of Computer Science
Naval Postgraduate School
Monterey, CA 93943

6. Genelkurmay Baskanligi.....1
Personel Baskanligi
Bakanliklar
Ankara, TURKEY

7. KARA KUVVETLERI KOMUTANLIGI..... 1
Personel Daire Baskanligi
Bakanliklar
Ankara, TURKEY

8. KARA KUVVETLERI KOMUTANLIGI.....1
Kutuphanesi
Bakanliklar
Ankara, TURKEY

9. Kara Harp Okulu.....1
Kutuphanesi
Dikmen
Ankara, TURKEY

10. Hanceri Sayat.....1
GnKur. MUBILDES.K.ligi.
Balgat
Ankara, TURKEY
11. LT Hanifi Aydin.....1
NPS 2 University Cir.
SGC # 2627
Monterey CA 93943
12. Mr. Steve Lose..... 1
Program Executive Officer, Submarines
PMS 450T2, NC2 5W64
2531 Jefferson Davis Highway
Arlington, VA 22242-5168
13. Mr. Gary Lacombe.....1
171 Branch Hill Rd
Preston, CT 06365